1 Essential Information

Overview

If you take the time to read anything in Signalogic DSP Software documentation, read the Essential Information section. All subjects are presented from a system-level perspective, with the idea that they might be read before the software is used, or at least while it is still being configured. A "development hierarchy" of Signalogic DSP Software explaining software component relationships is presented first, followed by a summary of individual software package capabilities. Other important sections include System Requirements, Performance Considerations, and DSP/Analog Hardware Considerations.

Main Topics

Relationship of Signalogic DSP Software Components

Hypersignal®-Macro Series Software Summary

DSPower®-HWLib Software Summary

DSP Source Code Interface Software Summary

DSPower-Block Diagram Software Summary

DSPower-C/C++ Code Generator Software Summary
Real-Time Convolver™ Summary
System Requirements
Memory Considerations
Performance Considerations
Video and Display Considerations
System Drivers and Resident Program Considerations
DSP/Analog Hardware Considerations
Technical Support

1.1 Relationship of Signalogic DSP Software Components

There are three main relationships between Signalogic DSP Software components:

1) Functional superset or subset, where there exists a version of a component that is a superset or subset of another in some way. Examples include real-time capability, and specific application capability.

2) DSP Engine, a hierarchical relationship where one component can make programmatic and transparent use of another, and overall functionality and convenience is available to the user by dealing only with the user-interface of the controlling component.

3) Source code generation capability, where one or more types of source code generation is not included in the standard version of a component, for marketing and cost reasons.

Functional Supersets and Subsets

Currently, functional supersets and subsets in Signalogic DSP Software components apply only to Hypersignal-Macro series software. These relationships are explained below:

1) Hypersignal-Macro Non-Real-Time. This package is a subset of Hypersignal-Macro. It contains the same simulation, file acquisition, waveform file displays, and DSP and math functions as Hypersignal-Macro, but has no real-time capability. Instruments, function acceleration using the DSP, anything else hardware-related is not active.

2) Hypersignal-Macro EX is a superset of Hypersignal-Macro. It is different only in that it contains additional macro language programs and functions.

3) Hypersignal-Acoustic is a superset of Hypersignal-Macro EX, containing three levels of increased functionality:

- additional macro language programs and functions
- additional intrinsic functionality inside displays and instruments
- additional device drivers for specialized acoustic/audio hardware

4) Nihon versions of Hypersignal packages. These versions are configured to run in DOS mode on NEC PC98xx series machines commonly found in Japan.

For more information, see section 1.2.1, Hypersignal-Macro Series Software Summary.

Related Topics

Functional Supersets and Subsets

DSP Engines

DSPower and DSP Engines

User Mode and Engine Mode

Programs, Template Files, and Function Files

Pipe.386

DSP Engines

Mainly due to the critical need for testability in increasingly complex systems, Signalogic's DSP software has evolved in a hierarchical relationship, with mature, reliable, off-the-shelf packages playing an important role.

The idea of using existing, off-the-shelf software packages in their object form--used as they are documented except controlled programmatically--has led to development of the "DSP Engine" concept. General application of this concept allows source code to be generated for not only Signalogic off-the-shelf software packages, but also other software packages, such as MATLAB® 4.x for Windows, and supported DSP/analog boards. Theoretically, the only requirement is that the engine be able to accept and run either source or object versions of programs. Of course, practical application is determined by considerations such as the type and popularity of language used, availability of development tools, etc.

Use of the DSP Engine concept also makes the overall system easier for users to understand and debug. Also, upgrading from one level to another is straightforward and cost-effective, since previous packages are still useful. And most importantly, the concept makes expansion of system capabilities simpler and more reliably implemented and tested for Signalogic. See also DSPower and DSP Engines.

Related Topics

Functional Supersets and Subsets

DSP Engines

DSPower and DSP Engines

User Mode and Engine Mode

Programs, Template Files, and Function Files

Pipe.386

DSPower and DSP Engines

[@j external application, controlling application, external controlling application]

The DSPower series of packages embodies a division-of-labor concept that relies on separate, well-defined "DSP Engines" to perform basic DSP, math, and hardware tasks. DSP Engines are expected to be off-the-shelf packages; in some cases, they may alternatively be able to run stand-alone, with their own user-interface (user mode operation).

In the DSPower-Block Diagram environment, user-interface and source code generation are performed by DSPower, while at least one or more DSP engines and various support tools are invoked remotely and transparently to achieve compilation and execution of generated code, as well as interactive operation. Signalogic believes that this represents a fundamental design advantage over other packages that perform simulations internally, and then generate source code as a separate process--which can produce inconsistent results and debug-resistant problems. Several system simplifications also result from the DSPower architecture; for example, block-diagrams are saved and retrieved in source code form. Currently there are three (3) DSP engine possibilities: MATLAB (version 4.x for Windows, available from The MathWorks), Hypersignal-Macro, Hypersignal-Macro EX, or Hypersignal-Acoustic (version 4.x for DOS, available from Hyperception or Signalogic), and a supported PC-compatible DSP/analog board from one of several manufacturers. Both MATLAB and Hypersignal-Macro offer superb simulation, by virtue of their rich procedural languages containing many functions and displays, and built-in compilers.

For example, the DSPower-Block Diagram package can instruct Hypersignal package (v4.0 or higher), MATLAB (v4.0 or higher), or a DSP/analog hardware platform to do useful work, under remote control, and transparently to the user. DSP Engines are responsible for the actual implementation of DSP/math functions, displays / visualizations / instruments, digital filter design, waveform file manipulation, etc. In the case of a supported DSP/analog hardware platform, it may be possible to execute complete block diagrams by downloading the compiled and linked form of DSPower-generated C code.

As another example, in the DSPower-HWLib DLL and DSPower-FileLib architectures, these DLLs communicates with a VxD (virtual device driver), which communicates directly with hardware drivers, waveform file functions, and DSP/math functions located in a DSP Engine. Engine examples include accessing the hardware drivers inside a Hypersignal-Macro series package (real-time versions of

which include drivers for over 60 types of DSP/analog hardware), or accessing DSP/math functions inside MATLAB from a C/C++ program. DSP Engine communication is achieved both by creating native DSP Engine programming language source code (e.g. Hypersignal macro language source, or .m file source), by sending programming via the VxD, or direct Windows messaging.

Related Topics

Functional Supersets and Subsets

DSP Engines

DSPower and DSP Engines

User Mode and Engine Mode

Programs, Template Files, and Function Files

Pipe.386

User Mode and Engine Mode

[@j simulated keyboard, user mode, engine mode]

Software applications that can be used as DSP Engines by DSPower-Block Diagram can operate in one of two *operating modes: user mode* or *engine mode*. In user mode, the application has been invoked manually, and is under control of an operator. In engine mode, the software application has been invoked programmatically, and is under the control of an outside Windows application (external controlling application; see DSP Engines, and DSPower and DSP Engines, above). In engine mode, visual aspects of the engine's user-interface are typically suppressed, and input is supplied to the engine in two main methods:

1. Through generation of an initial program instructing the engine. To do this the controlling application generates source code in the native language for the engine, compiles it (or submits it to the engine for compiling), and gives it to the engine for execution. Examples include Hypersignal macro language and MATLAB .m files.

2. By supplying input to the engine as if coming from a "simulated keyboard", which allows operation of the engine to be modified and

controlled dynamically, in complement to, or as a replacement for, the initial program given to the engine.

Normally, a description of input required to control a DSP Engine function will be accurate for both user mode and engine mode, except the input must be presented to the DSP Engine in a different manner. For example, in the case of Hypersignal-Macro series software packages, the macro language in these packages has a simulated keyboard input facility, which can accept input from the actual keyboard, the macro language program (interactive commands; see Hypersignal-Macro Series Users Manual section 7.3.2, Issuing Interactive Commands From Macros), or, when Hypersignal is operating in engine mode, from an external application using the Pipe.386 VxD. This makes it possible for external applications to control all aspects of Hypersignal-Macro series software operation using debugged, well-documented input commands.

In some cases, though, certain types of input in user mode may not apply in engine mode. This can occur because some part of the user interface, or some visual cue, is not available in engine mode, or more often, because in engine mode the input is taken care of automatically. When this occurs in decriptions of Hypersignal-Macro series software in Signalogic DSP Software documentation, the section or heading is marked clearly with "In User Mode", to highlight the expectation that the section only pertains if the software is being operated in user mode.

Related Topics

Functional Supersets and Subsets

DSP Engines

DSPower and DSP Engines

User Mode and Engine Mode

Programs, Template Files, and Function Files

Pipe.386

Programs, Template Files, and Function Files

[@j program files, template files, function files, reduced architecture]

The DSP Engine concept naturally leads to a division of native language source code generated by DSPower series software and submitted to DSP Engines into two types:

1. programs
2. functions

At run-time, programs are the text file equivalent of an onscreen block or block diagram, representing some DSP, math, display, instrument, etc. operation, or combination of operations, to be performed. Programs make one or more calls to functions. Because there are many blocks in the DSPower system, and potentially an unlimited number of block diagrams, the actual code content of programs must be kept to a minimum.

In the DSPower-Block Diagram architecture, "template files" are used as the basis for creating source code representing ready-to-execute DSP Engine programs, which then call "function files". Much of the content of template files is non-source code information: block registration, connection definitions, parameter descriptions, graphical control interface descriptions, current onscreen status, etc.

Functions comprise the bulk of "working code" in the DSPower architecture. They are not typically altered by DSPower series software, which means they can exist in the system in either source or executable form. A given function might be invoked by many different programs. DSPower-Block Diagram software maintains libraries of function files which represent the basic capabilities of the system, and tend not to overlap.

One major benefit of the program / function subdivision is to allow tasks being executed from either user mode or engine mode to share the bulk of the source code needed (function files), to reduce the possiblility of error or differences in results. The same function files are being used, although the user mode program which does the calling may look much different than the DSPower template file. See User Mode and Engine mode, above.

Another benefit of the program / function subdivision is to reduce the actual amount of source code in the overall system by creating a fundamental, orthogonal set of function files that can be called in various combinations or sequences to do just about any task of which DSP Engines are capable.

The overall result of the DSPower architecture is a system both easy for the user to understand, and for Signalogic to maintain.

Related Topics

Functional Supersets and Subsets

DSP Engines

DSPower and DSP Engines

User Mode and Engine Mode

Programs, Template Files, and Function Files

Pipe.386

Pipe.386

The Pipe.386 VxD (virtual device driver) is included with DSPower series software components, allowing them to communicate with DOS DSP Engines such as Hypersignal-Macro and Hypersignal-Acoustic.

The VxD concept in Windows is somewhat of a "magic key", allowing system communication and operation that is not normally documented by Microsoft or otherwise obvious. In the case of DSPower, Pipe.386 is used to send interactive macro language instructions and receive processing and display results from the macro language "simulated keyboard" entry facility in Hypersignal packages. Use of a VxD to perform this communication means that it is ultrafast, or at least as fast as in possible in Windows, and not subject to protocol and other restrictions of a method such as DDE.

Use of a VxD allows one or more Hypersignal packages to be "active" and performing tasks designated by a DSPower series software package, yet not visible from the DSPower user-interface.

Related Topics

Functional Supersets and Subsets
DSP Engines
DSPower and DSP Engines
User Mode and Engine Mode
Programs, Template Files, and Function Files
Pipe.386

1.2 Summary of Signalogic DSP Software Components

Summaries of major Signalogic DSP Software components are presented to help answer questions such as "what can it do?" and "which package do I need?". These sections also help explain the relationship between packages. See Appendix A for a development hierarchy chart that helps explain the relationship between Signalogic DSP Software components, and other off-the-shelf packages with which they are compatible.

1.2.1 Hypersignal®-Macro Series Software Summary

The Hypersignal-Macro series of software packages includes Hypersignal-Macro NRT, Hypersignal-Macro, Hypersignal-Macro EX, and Hypersignal-Acoustic. There are also "Nihon" versions of these packages modified to run on NEC PC98xx-compatible machines.

Hypersignal-Macro is a DOS program with menu-driven and programming language user interfaces; it offers math/DSP simulation, instrumentation, filter design, data acquisition, and display and interactive analysis. Hypersignal-Macro can also operate as a "DSP Engine" with DSPower, in which case it is controlled transparently by DSPower. Hypersignal-Macro supports more than 60 types of DSP/analog, acquisition, and multimedia hardware.

Hypersignal-Macro EX is a superset of Hypersignal-Macro which includes additional functions such as long FFT, convolution, and correlation, frequency zoom, utility macros, and more.

Hypersignal-Macro NRT is a "non-real-time" version of Hypersignal-Macro, which offers the same simulation and file-based processing function, but no real-time support. The package contains no device drivers for DSP/acquisition hardware; all input data is expected to be imported using the built-in File Acquisition function, which is capable of gathering data from a wide variety of binary and ASCII file formats, with control over data precision, header filtering, multichannel, complex, and other format characteristics. Hypersignal-Macro NRT is typically available for 1/2 the cost of Hypersignal-Macro.

Hypersignal-Acoustic is a superset of Hypersignal-Macro EX, and includes additional functions such as MLS stimulus & response, equalization, Schroeder integration, minimum phase calculation, reverse waterfall, additional Spectrum Analyzer and waveform display capability, device drivers for TEF20HI, DSP-16, ProPort/DatPort, and Tahiti/Monterey.

NOTE: In the software function overviews below, "Hypersignal software" is sometimes used as shorthand to mean "Hypersignal-Macro series software".

Below is a list of functions, displays, instruments, and utilities inside Hypersignal series software packages, as well as system consideration topics.

Hypersignal Functions

DSP Simulation Function Overview

FFT Generation

Convolution/Correlation

LPC Autocorrelation

Recursive Filtering

Power Spectra Generation

Transfer Function Generation

Inverse FFT Generation

Difference Equations

Sampling Rate Conversion

Hypersignal-Macro EX Functions

Long FFT

Long Convolution/Correlation

Frequency Zoom

Minimum Phase

Wavelet Transform

Hypersignal-Acoustic Functions

Schroeder Integration

Equalization

MLS Stimulus & Response

Hypersignal Digital Filter Design

Digital Filter Design Overview

FIR Filter Design

IIR Filter Design

Filter DSP Code Generation

Pole-Zero Display

File-Based Filter Simulation

"Snap-In" Real-Time Filtering

Hypersignal Interactive Display

Interactive Display Overview

Time Domain Waveform Display/Edit

Frequency Domain Magnitude and Phase Display

Joint Time-Frequency Waterfall Display

Joint Time-Frequency Contour Display

Filter Design Display

Pole-Zero Display

Hypersignal Instrumentation & Measurement

Instrumentation & Measurement Overview

Spectrum Analyzer

Digital Oscilloscope

Continuous Record/Output To/From Disk

Stimulus & Response

Signal Synthesis

Hypersignal Macro Language

Macro Language Overview

Macro Record Mode

Macro Programs: Ready-to-Execute Functions

Creating Macro Programs and Functions

Macro Language Variables, Constructs, and Procedures

Hypersignal File Acquisition/Conversion Functions

File Import/Export Functions

Tim2Wav and Wav2Tim Waveform File Conversion

Hypersignal External Program Interface

External Program Control

Hypersignal System Considerations

User-Interface

Online Help

Graphics and Mouse Support

80x87 Math Coprocessor Support

Printer Support

Documentation

Hypersignal DSP Simulation Function Overview

Hypersignal-Macro series software DSP simulation functions accept input in time or frequency domain waveform file format, producing time or frequency domain output waveform files. Input time domain file formats can be Hypersignal .TIM or Microsoft .WAV. Output time domain file formats are in Hypersignal .TIM format for most functions; additional Hypersignal-Macro EX functions allow direct .WAV format output. A tim2wav format conversion function is included with all Hypersignal packages. All frequency domain output waveform files are in Hypersignal formats. (For more information about waveform file formats, see Signalogic DSP Software Reference Guide sections 4.1 and 4.2, Hypersignal Waveform File Format, and Microsoft .WAV Waveform File Format).

FFT Generation in Hypersignal

[@k transform,, Fast Fourier]

All Hypersignal-Macro series software packages contain a built-in complex input Fast Fourier Transform (FFT) function. In addition, when operating in user mode, the Hypersignal-Macro EX and Hypersignal-Acoustic packages offer a lfft.mac program which allows a "long FFT" function to be invoked from the Macro Manager menu.

Both the built-in FFT function and the f_fft.mac long FFT function (which is called by the lfft.mac program), can be called from outside Hypersignal, when operating in engine mode.

The built-in FFT function can accept input waveform, framesize, FFT length, window (Hamming, Hanning, Blackman, Bartlett, Gaussian, rectangular, user-defined), and overlap as input parameters. Output magnitude and phase waveform data are generated always; output real and imaginary data can optionally be generated for complex frequency domain operations. For the built-in FFT function, actual FFT

processing may be performed either by the host PC or by DSP hardware currently selected as a "DSP Accelerator".

Input waveform size is not limited, and can be segmented according to framesize and overlap selections.

Current maximum FFT length is 4096 pt. for the built-in version (may be less for some DSP hardware). Current maximum length for the lfft.mac long FFT function, is 262,144, although it may be less depending on available host PC system memory (see Signalogic DSP Software System Guide section 1.3, System Requirements).

Additional Hypersignal Functions

FFT Generation in Hypersignal

Convolution/Correlation in Hypersignal

LPC Autocorrelation in Hypersignal

Recursive Filtering in Hypersignal

Power Spectra Generation in Hypersignal

Transfer Function Generation in Hypersignal

Inverse FFT Generation in Hypersignal

Difference Equations in Hypersignal

Sampling Rate Conversion in Hypersignal

Hypersignal-Macro EX Functions

Long FFT in Hypersignal

Long Convolution/Correlation in Hypersignal

Frequency Zoom in Hypersignal

Minimum Phase in Hypersignal

Wavelet Transform in Hypersignal

Hypersignal-Acoustic Functions

Schroeder Integration in Hypersignal

Equalization in Hypersignal

Convolution/Correlation in Hypersignal

[@k circular correlation, crosscorrelation, autocorrelation]

All Hypersignal-Macro series software packages contain built-in convolution, cross-correlation, circular correlation, and autocorrelation functions. In addition, when operating in user mode, the Hypersignal-Macro EX and Hypersignal-Acoustic packages offer lconv.mac and lcorr.mac programs which allow "long convolution" and "long correlation" functions to be invoked from the Macro Manager menu.

Both the built-in Convolution/Correlation function and the f_conv.mac long function (which is called by the lconv.mac and lcorr.mac programs), can be called from outside Hypersignal, when operating in engine mode.

Both the built-in functions and long versions can accept input waveform, filter waveform, and output waveform as input parameters. Filter waveforms can be the result of standard Hypersignal functions such as FIR and IIR Filter Construction and Spectrum Analyzer transfer function storage, or any arbitrary time domain waveform. For the built-in Convolution/Correlation function, actual processing may be performed either by the host PC or by DSP hardware currently selected as a "DSP Accelerator".

Current maximum filter waveform size is 16384 pt. (single-channel data), and 8192 pt. (two-channel data) for the built-in functions. Current maximum filter lengths for the lconv.mac and lcorr.mac long functions is about 180,000 pts. for single-channel data, although it may be less depending on available host PC system memory (see Signalogic DSP Software System Guide section 1.3, System Requirements, for system RAM requirements with Hypersignal-Macro EX). Input waveform size is not limited.

Additional Hypersignal Functions

FFT Generation in Hypersignal

Convolution/Correlation in Hypersignal

LPC Autocorrelation in Hypersignal
Recursive Filtering in Hypersignal
Power Spectra Generation in Hypersignal
Transfer Function Generation in Hypersignal
Inverse FFT Generation in Hypersignal
Difference Equations in Hypersignal
Sampling Rate Conversion in Hypersignal

Hypersignal-Macro EX Functions

Long FFT in Hypersignal
Long Convolution/Correlation in Hypersignal
Frequency Zoom in Hypersignal
Minimum Phase in Hypersignal
Wavelet Transform in Hypersignal

Hypersignal-Acoustic Functions

Schroeder Integration in Hypersignal
Equalization in Hypersignal
MLS Stimulus & Response in Hypersignal

LPC Autocorrelation in Hypersignal

[@k autocorrelation,, LPC, linear predictive coding]

All Hypersignal-Macro series software packages contain a built-in LPC (Linear Predictive Coding) Autocorrelation function.

LPC autocorrelation is often used on periodic or semiperiodic data to construct a series of filters with frequency responses that represent the major frequency domain energy areas in the input data. The input data is segmented into frames, and a filter is constructed on each frame of input data, to allow for time-varying characteristics in the input data. The resulting series of "analysis filter" coefficients can be used to create a series of impulse responses, which, when strung

together, represent a "facsimile" of the input data. A form of compression is inherent in this process, because only the filter coefficients are needed to represent the original input data, with the overall compression ratio being the input framesize divided by the number of coefficients. And, the resulting series of frequency response frames, resulting from either the filter coefficients directly or the synthesized data is "smooth", showing only the major energy areas.

LPC autocorrelation function input paramters include input waveform, framesize, filter length, window (Hamming, Hanning, Blackman, Bartlett, Gaussian, rectangular, user-defined), and output waveform. Either analysis or reflection filter coefficients may be chosen. Maximum filter length is 31; input waveform size is not limited.

Additional Hypersignal Functions

FFT Generation in Hypersignal

Convolution/Correlation in Hypersignal

LPC Autocorrelation in Hypersignal

Recursive Filtering in Hypersignal

Power Spectra Generation in Hypersignal

Transfer Function Generation in Hypersignal

Inverse FFT Generation in Hypersignal

Difference Equations in Hypersignal

Sampling Rate Conversion in Hypersignal

Hypersignal-Macro EX Functions

Long FFT in Hypersignal

Long Convolution/Correlation in Hypersignal

Frequency Zoom in Hypersignal

Minimum Phase in Hypersignal

Wavelet Transform in Hypersignal

Hypersignal-Acoustic Functions

Schroeder Integration in Hypersignal

Equalization in Hypersignal

MLS Stimulus & Response in Hypersignal

Recursive Filtering in Hypersignal

[@k filtering,, IIR, filtering,, recursive]

All Hypersignal-Macro series software packages contain a built-in recursive filtering function, which can apply IIR filters in factored or direct form to input time domain waveform data.

Recursive filtering function input parameters include input waveform, filter coefficient file, and output waveform. Input filter coefficient files are in ASCII format and can be standard output from the Hypersignal Filter Construction functions, or user-defined. The coefficient file format accepts both factored and direct-form numerator (poles) and denominator (zeros), and can be applied in "biquad cascade" (factored) form or direct form.

Additional Hypersignal Functions

FFT Generation in Hypersignal

Convolution/Correlation in Hypersignal

LPC Autocorrelation in Hypersignal

Recursive Filtering in Hypersignal

Power Spectra Generation in Hypersignal

Transfer Function Generation in Hypersignal

Inverse FFT Generation in Hypersignal

Difference Equations in Hypersignal

Sampling Rate Conversion in Hypersignal

Hypersignal-Macro EX Functions

Long FFT in Hypersignal

Long Convolution/Correlation in Hypersignal

Frequency Zoom in Hypersignal

Minimum Phase in Hypersignal

Wavelet Transform in Hypersignal

Hypersignal-Acoustic Functions

Schroeder Integration in Hypersignal

Equalization in Hypersignal

MLS Stimulus & Response in Hypersignal

Power Spectra Generation in Hypersignal

[@k spectra estimation]

All Hypersignal-Macro series software packages contain a built-in frequency domain function which estimates the power spectra from a magnitude waveform created by previous FFT Generation.

Power Spectra Generation input parameters include input waveform filename, method of estimation, and output waveform. The method of estimation can be "group averaging", where the input waveform is segmented into one or more sections which are averaged together independently, creating that number of output frames, "running averaging", where a specified number of frames from the input waveform are averaged together, producing a new output frame each time the "boxcar" increments one frame, and "exponential averaging", where a one-pole lowpass filter (with selectable time constant) is applied between input frames.

Additional Hypersignal Functions

FFT Generation in Hypersignal

Convolution/Correlation in Hypersignal

LPC Autocorrelation in Hypersignal

Recursive Filtering in Hypersignal

Power Spectra Generation in Hypersignal

Transfer Function Generation in Hypersignal
Inverse FFT Generation in Hypersignal
Difference Equations in Hypersignal
Sampling Rate Conversion in Hypersignal

Hypersignal-Macro EX Functions

Long FFT in Hypersignal
Long Convolution/Correlation in Hypersignal
Frequency Zoom in Hypersignal
Minimum Phase in Hypersignal
Wavelet Transform in Hypersignal

Hypersignal-Acoustic Functions

Schroeder Integration in Hypersignal
Equalization in Hypersignal
MLS Stimulus & Response in Hypersignal

Transfer Function Generation in Hypersignal

[@k deconvolution]

All Hypersignal-Macro series software packages contain a built-in frequency domain function which calculates the complex ratio of two waveforms presumably representing the output and input of an "unknown system".

Input parameters include input and output waveform filenames. Output magnitude, phase, and complex waveforms are produced.. The complex output can be used to calculate the equivalent time domain impulse response.

Transfer function analysis is often used to characterize unknown systems when both the input and output of the system are unknown (as opposed to stimulus & response measurement, when the input is

known; see below). Assuming that the spectral content of the input was wideband, or was wideband over a period of time due to averaging, the magnitude result can represent the frequency response of the unknown system

Additional Hypersignal Functions

FFT Generation in Hypersignal

Convolution/Correlation in Hypersignal

LPC Autocorrelation in Hypersignal

Recursive Filtering in Hypersignal

Power Spectra Generation in Hypersignal

Transfer Function Generation in Hypersignal

Inverse FFT Generation in Hypersignal

Difference Equations in Hypersignal

Sampling Rate Conversion in Hypersignal

Hypersignal-Macro EX Functions

Long FFT in Hypersignal

Long Convolution/Correlation in Hypersignal

Frequency Zoom in Hypersignal

Minimum Phase in Hypersignal

Wavelet Transform in Hypersignal

Hypersignal-Acoustic Functions

Schroeder Integration in Hypersignal

Equalization in Hypersignal

MLS Stimulus & Response in Hypersignal

Inverse FFT Generation in Hypersignal

[@k FFT,, inverse, transform,, Fast Fourier]

All Hypersignal-Macro series software packages contain a built-in complex input inverse Fast Fourier Transform (iFFT) function. In addition, when operating in user mode, the Hypersignal-Macro EX and Hypersignal-Acoustic packages offer a lifft.mac program which allows a "long iFFT" function to be invoked from the Macro Manager menu.

Both the built-in iFFT function and the f_ifft.mac long iFFT function (which is called by the lifft.mac program), can be called from outside Hypersignal, when operating in engine mode.

The built-in iFFT function can accept input waveform filename as an input parameter. Output is a time domain waveform. For the built-in iFFT function, actual iFFT processing may be performed either by the host PC or by DSP hardware currently selected as a "DSP Accelerator".

Input waveform size is not limited.

Current maximum FFT length is 4096 pt. for the built-in version (may be less for some DSP hardware). Current maximum length for the f_ifft.mac long iFFT function, is 262,144, although it may be less depending on available host PC system memory (see Signalogic DSP Software System Guide section 1.3, System Requirements).

Additional Hypersignal Functions

FFT Generation in Hypersignal

Convolution/Correlation in Hypersignal

LPC Autocorrelation in Hypersignal

Recursive Filtering in Hypersignal

Power Spectra Generation in Hypersignal

Transfer Function Generation in Hypersignal

Inverse FFT Generation in Hypersignal

Difference Equations in Hypersignal

Sampling Rate Conversion in Hypersignal

Hypersignal-Macro EX Functions

Long FFT in Hypersignal

Long Convolution/Correlation in Hypersignal

Frequency Zoom in Hypersignal

Minimum Phase in Hypersignal

Wavelet Transform in Hypersignal

Hypersignal-Acoustic Functions

Schroeder Integration in Hypersignal

Equalization in Hypersignal

MLS Stimulus & Response in Hypersignal

Difference Equations in Hypersignal

[@k equations,, difference]

All Hypersignal-Macro series software packages contain a Difference Equations function which allows generation of new waveforms and manipulation of existing waveforms using various operators and math functions, specified in classic difference equation form. Input sources include existing waveforms, unit-step and delta functions, and previous output (recursive) data. Operators include signal arithmetic, multiplication, division (and integer division) modulus, differentiation, integration, min and max, and a frequency domain "frame operator". Functions include several trigonometric and transcendental functions, random noise, and various linear and non-linear math functions.

With the Hypersignal Difference Equations, it is straightforward to build sine, pulse, noise, etc. function generators, as well as to operate arbitrarily on existing waveforms. Equations are based on a discrete time index "n", which begins at zero and increments a specified number of times, according to user entry. The difference equation evaluates at each discrete time index. Note that it is also possible to write expressions for n, both when used as an input index, and also when used as an output index, on the left side of the equals sign.

Several example equations are shown below. Consider the following equation, which calculates a running average:

y[n] = y[n–1] + x1[n] – x1[n–Fr]

The equation below would generate a sine waveform:

y[n] = sin(2∗pi∗n∗Fc/Fs)

where "Fs" refers to the desired sampling frequency, and "Fc" to the desired sine wave frequency. This equation can be generalized to a function generator that describes one period of the desired equation (by replacing "F(n)" below with the appropriate expression), and then replicates the initial period as many times as necessary:

y[n] = F(n)∗(u[n] – u[n–M]) + y[n–M]

In the above equation, M specifies the period in samples. A simple example would be to replace F(n) with n, which would create a sawtooth generator. The equation below would "spectrally invert" a time domain waveform, causing dc to be exchanged for half the sampling rate:

y[n] = x1[n]∗(–1)^n

Finally, consider multiplying a time domain waveform by a complex exponential to produce an in-phase real component and a quadrature imaginary component, where Fc=cutoff frequency, and Fs=sampling frequency:

y[n] = x1[n]∗cos[2∗pi∗Fc∗n/Fs] – j∗x1[n]∗sin[2∗pi∗Fc∗n/Fs]

When sample manipulation, by itself or in combination with signal mathematics, is required, the method is equally simple and elegant. The equation below would delay a waveform by "M" samples:

y[n] = x1[n–M]

The equation below treats x1 as a circular buffer, moving samples an amount specified by M, where Len is the length of x1 and "%" indicates modulus division:

y[n] = x1[(n+M)%Len]

Additional Difference Equations Topics

Input Sources in Difference Equations

Arithmetic Operators in Difference Equations

Non-Arithmetic Operators in Difference Equations

Constants in Diffference Equations

Trig, Transcendental, and Calculus Functions in Difference Equations

Additional Hypersignal Functions

FFT Generation in Hypersignal

Convolution/Correlation in Hypersignal

LPC Autocorrelation in Hypersignal

Recursive Filtering in Hypersignal

Power Spectra Generation in Hypersignal

Transfer Function Generation in Hypersignal

Inverse FFT Generation in Hypersignal

Difference Equations in Hypersignal

Sampling Rate Conversion in Hypersignal

Hypersignal-Macro EX Functions

Long FFT in Hypersignal

Long Convolution/Correlation in Hypersignal

Frequency Zoom in Hypersignal

Minimum Phase in Hypersignal

Wavelet Transform in Hypersignal

Hypersignal-Acoustic Functions

Schroeder Integration in Hypersignal

Equalization in Hypersignal

MLS Stimulus & Response in Hypersignal

Input Sources in Difference Equations

[@k difference equations input sources]

Below is a partial list of possible input sources ("exp" represents the discrete time index, as explained in the main discussion of Difference Equations, and can be an expression of the sample index "n" or the frame index "m"):

Table 1 List of Difference Equations Input Sources

x1[exp]	refers to waveform specified by x1 INPUT FILENAME field;
x2[exp]	refers to waveform specified by x2 INPUT FILENAME field;
y[exp]	refers to previous output, sometimes called recursive input or feedback
u[exp]	unit-step function; accesses to this array return 0 if exp < 0 and 1 if exp ≥>= 1
d[exp]	delta function; accesses to this array return 0 if exp ≠<> 0 and 1 if exp = 0

Additional Difference Equations Topics

Input Sources in Difference Equations

Arithmetic Operators in Difference Equations

Non-Arithmetic Operators in Difference Equations

Constants in Diffference Equations

Trig, Transcendental, and Calculus Functions in Difference Equations

Arithmetic Operators in Difference Equations

[@k operators,, difference equations, difference equations operators]

Table 2 Difference Equations Arithmetic Operators

*	multiply
/	divide
\	integer divide
%	modulus
+	add
–	subtract

Additional Difference Equations Topics

Input Sources in Difference Equations

Arithmetic Operators in Difference Equations

Non-Arithmetic Operators in Difference Equations

Constants in Diffference Equations

Trig, Transcendental, and Calculus Functions in Difference Equations

Non-Arithmetic Operators in Difference Equations

[@k operators,, difference equations, difference equations operators]

Table 3 Difference Equations Non-Arithmetic Operators

<, >	The "<" and ">" characters are MIN and MAX operators, respectively
,	The frame operator. A comma causes operands to be read as "frame, offset"

Additional Difference Equations Topics

Input Sources in Difference Equations

Arithmetic Operators in Difference Equations

Non-Arithmetic Operators in Difference Equations

Constants in Diffference Equations

Trig, Transcendental, and Calculus Functions in Difference Equations

Constants in Difference Equations

[@k difference equations constants]

Table 4 Difference Equations Constants

pi	returns πpi, where πpi = 3.14159265...
Fs	returns value of SAMPLING FREQUENCY field
Fr	returns value of FRAMESIZE field
j	sqrt(−1)√−1

Additional Difference Equations Topics

Input Sources in Difference Equations

Arithmetic Operators in Difference Equations

Non-Arithmetic Operators in Difference Equations

Constants in Diffference Equations

Trig, Transcendental, and Calculus Functions in Difference Equations

Trig, Transcendental, and Calculus Functions in Difference Equations

[@k function,, difference equations, difference equations functions]

Table 5 Difference Equations Functions

ABS(arg)	|arg|
ACOS(arg)	returns, in radians, arccosine of arg

ASIN(arg)	returns, in radians, arcsine of arg
ATAN(arg)	returns, in radians, arctangent of arg
COS(arg)	cosine of arg; arg expected in radians
COT(arg)	cotangent of arg; arg expected in radians
Darray[exp]	returns derivative of array[exp] value
EXP(arg)	*e***arge^{arg}, where *e* = 2.71828182..
FRAC(exp)	returns fractional part of real value
larray[exp]	returns integral of array[exp] value
INT(arg)	removes fractional part of real value
LOG(arg)	log10(arg)$\log_{10}$(arg)
LN(arg)	ln(arg)
RAN(arg)	returns pseudo-random real number between 0 and arg, inclusive
RND(arg)	rounds arg to nearest integer
ROUND(arg)	rounds arg to nearest integer
SIGN(arg)	returns –1 for arg < 0, returns 1 for arg ≥>= 0
SIN(arg)	sine of arg; arg expected in radians
SQR(arg)	arg^2
SQRT(arg)	sqrt(arg)√arg
TAN(arg)	tangent of arg; arg expected in radians
TENX(arg)	10**arg10^{arg}
TRUNC(arg)	truncates arg to next lowest integer

Additional Difference Equations Topics

Input Sources in Difference Equations

Arithmetic Operators in Difference Equations

Non-Arithmetic Operators in Difference Equations

Constants in Difference Equations

Sampling Rate Conversion in Hypersignal

[@k downsampling, conversion,, sampling rate]

The fsconv.mac program is included with all Hypersignal-Macro series software packages.

When operating Hypersignal in user mode, the fsconv.mac program allows sampling rate conversion to appear as a virtual function when invoked from the Macro Manager menu. The fsconv.mac program prompts for an input time domain waveform and stretches or contracts the waveform to produce an output time domain with a different sampling rate. Input parameters include "multiply" and "divide" integer values, whose ratio, when multiplied against the input waveform sampling frequency, equals the output waveform sampling frequency. For example, to reduce an input waveform sampling rate by a factor of 10, a multiply integer of 1 should be entered, and a divide integer of 10. To increase sampling rate by a factor of 2, a multiply integer of 2 should be entered, and a divide integer of 1. See also comments in the fsconv.mac and dsplib.inc files.

The fsconv.mac program calls the f_fsconv.mac function, which can also be called from an outside application when Hypersignal is being operated in engine mode.

Additional Hypersignal Functions

FFT Generation in Hypersignal

Convolution/Correlation in Hypersignal

LPC Autocorrelation in Hypersignal

Recursive Filtering in Hypersignal

Power Spectra Generation in Hypersignal

Transfer Function Generation in Hypersignal

Inverse FFT Generation in Hypersignal

Difference Equations in Hypersignal

Sampling Rate Conversion in Hypersignal

Hypersignal-Macro EX Functions

Long FFT in Hypersignal

Long Convolution/Correlation in Hypersignal

Frequency Zoom in Hypersignal

Minimum Phase in Hypersignal

Wavelet Transform in Hypersignal

Hypersignal-Acoustic Functions

Schroeder Integration in Hypersignal

Equalization in Hypersignal

MLS Stimulus & Response in Hypersignal

Frequency Zoom in Hypersignal

[@k zoom,, frequency, baseband zoom,, frequency]

The fzoom.mac program is included only with Hypersignal-Macro EX and Hypersignal-Acoustic software.

When operating Hypersignal in user mode, the fzoom.mac program allows frequency zoom to appear as a virtual function when invoked from the Macro Manager menu. The fzoom.mac program prompts for an input time domain waveform, and translates a portion of its overall bandwidth to the baseband, creating an output time domain waveform. Start (f_1) and stop (f_2) frequencies must be specified, which are entered as values normalized to 1/2 the sampling rate, or $0 \leq <= f_1 \leq <= 1$ and $0 \leq <= f_2 \leq <= 1$. The output time domain waveform filename must also be specified. See also comments inside the fzoom.mac and dsplib.inc files.

The fzoom.mac program calls the f_fzoom.mac function, which can also be called from an outside application when Hypersignal is being operated in engine mode.

Additional Hypersignal Functions

FFT Generation in Hypersignal
Convolution/Correlation in Hypersignal
LPC Autocorrelation in Hypersignal
Recursive Filtering in Hypersignal
Power Spectra Generation in Hypersignal
Transfer Function Generation in Hypersignal
Inverse FFT Generation in Hypersignal
Difference Equations in Hypersignal
Sampling Rate Conversion in Hypersignal

Hypersignal-Macro EX Functions

Long FFT in Hypersignal
Long Convolution/Correlation in Hypersignal
Frequency Zoom in Hypersignal
Minimum Phase in Hypersignal
Wavelet Transform in Hypersignal

Hypersignal-Acoustic Functions

Schroeder Integration in Hypersignal
Equalization in Hypersignal
MLS Stimulus & Response in Hypersignal

Minimum Phase in Hypersignal

[@k phase,, minimum]

The minphase.mac program is included only with Hypersignal-Macro EX and Hypersignal-Acoustic software

When operating Hypersignal in user mode, the minphase.mac program allows minimum phase to appear as a virtual function when invoked from the Macro Manager menu. The minphase.mac program prompts for an input time domain waveform, which should be a filter response

or other impulse response, and creates a minimum phase version, also as a time domain waveform, with a user-specified output filename.

Minphase.mac attempts to minimize the overall group delay of the input impulse response, while leaving the frequency domain magnitude response unchanged. The minimum phase version should have the same frequency domain magnitude response (you can do an FFT Generation to check this), but minimized phase response, and an impulse response with energy concentrated near time=zero. For example, if you run minphase.mac on a Hypersignal FIR filter impulse response, you should see the resulting time domain impulse response looking somewhat like a "left-shifted" version of the original.

The minphase.mac program calls several macro language functions, including Difference Equations and FFT, during its course of processing. These other functions can also be called from an outside application when Hypersignal is being operated in engine mode.

When run on a waveform that already has minimum phase, such as an IIR filter impulse response, there should be essentially no change in the output from the input--in any of the magnitude, phase, or time domain impulse responses.

Examples of minimum phase usage include "what-if" scenarios to explore the effects of minimizing the delay in waveforms/sequences, reducing FIR filter lengths in real-time systems that are processing-intensive or have high sampling rates, and creating minimum phase versions of impulse responses that have been realized through equalization (transfer function measurement of physical channels).

Additional Hypersignal Functions

FFT Generation in Hypersignal

Convolution/Correlation in Hypersignal

LPC Autocorrelation in Hypersignal

Recursive Filtering in Hypersignal

Power Spectra Generation in Hypersignal

Transfer Function Generation in Hypersignal

Inverse FFT Generation in Hypersignal

Difference Equations in Hypersignal

Sampling Rate Conversion in Hypersignal

Hypersignal-Macro EX Functions

Long FFT in Hypersignal

Long Convolution/Correlation in Hypersignal

Frequency Zoom in Hypersignal

Minimum Phase in Hypersignal

Wavelet Transform in Hypersignal

Hypersignal-Acoustic Functions

Schroeder Integration in Hypersignal

Equalization in Hypersignal

MLS Stimulus & Response in Hypersignal

Wavelet Transform in Hypersignal

[@k transform,, wavelet, continuous wavelet transform]

The wlt.mac program is included with both Hypersignal-Macro EX and Hypersignal-Acoustic for those who want to experiment with it. This program is still considered to be in an alpha-state, with improvements continuing to be added.

When operating Hypersignal in user mode, the wlt.mac program allows continuous wavelet transform to appear as a virtual function when invoked from the Macro Manager menu. The wlt.mac program prompts for input waveform filename, wavelet specification, output waveform filename, and number of scale values.

The wlt.mac program calls the f_wlt.mac function, which can also be called from an outside application when Hypersignal is being operated in engine mode.

In either mode, the wavelet specification can be one of the following:

1. A complex wavelet file (enter filename).

2. A real wavelet file (enter filename). In this case, a complex wavelet is created by the program, using a Hilbert Transform technique.

3. Offset and length of input filename (enter in samples, or points). In this case, the wavelet is taken from signal itself. For example, an entry of '76,32' in this field would cause the program to use a 32-point segment, starting at sample 76, from the input waveform. Note: when using this option, it is suggested that a somewhat periodic segment of the input waveform is chosen.

Some notes about the f_wlt.mac function:

1. A center frequency of the wavelet is automatically calculated using an FFT. Periodic wavelets lend themselves more to this calculation.

2. A continuous wavelet transform is performed using a chirp-z transform method developed at the Univ. of Illinois and Rice University. See the Hypersignal macro language wlt.mac file for more details and algorithm/theory references and credits.

3. Output is produced compatible with Hypersignal frequency domain displays.

Currently, the function does not allow the input waveform to be segmented; as a result, large input waveforms are not yet practical.

One interesting observation to make using the wavsig.tim (input waveform) and gauss.tim (wavelet) files as examples in a comparison of the wavelet transform and a short time Fourier transform (STFT) computed using a baseband equivalent of the wavelet as a window function. This can be accomplished using the Gauss window option in the FFT Generation function (because the wavelet in this example is Gaussian in shape with a center frequency of $\pi/3$). For example, perform the following steps:

A description of the steps needed to make this comparison when operating Hypersignal in user mode is given below:

First, select the F3 key, which invokes the Macro Manager menu, and enter 'wlt' on the first line. This will cause a data-entry menu to appear prompting you for necessary entries. Use 'wavsig' as input waveform, 'gauss' as the wavelet filename, 'out' as the output filename, and number of scale values as '128'.

Second, select the FFT Generation menu item in Hypersignal and enter 'wavsig' as the input waveform. Enter the following parameters:

FFT order = 8 (FFT size = 256)

Framesize = 32 (samples)

Overlap = (Framesize−1)/Framesize∗100 (percent)

Window = Ga (Gaussian window)

Precision and Mag/Phase Approximation fields--if they should appear because you have no DSP hardware accelerator selected--as "don't cares" (they affect accuracy and speed somewhat, and can make a difference when exact results or large waveforms are required)

Third, enter 'wavsig' and 'out' in the 2-D Spectrograph display to see a comparison. You may want to enable the DUAL AUTOSCALE field in the User Setup area to see better comparison of amplitudes.

Overall, the choice of what type of analysis method to use (i.e. STFT, wavelet transform, chirp-z, etc.) is highly application- and data-dependent. Many experiments and iterations may be required to determine what works best. If indeed a wavelet transform method is to be used, further experimentation and research is needed to choose an effective wavelet.

Additional Hypersignal Functions

FFT Generation in Hypersignal

Convolution/Correlation in Hypersignal

LPC Autocorrelation in Hypersignal

Recursive Filtering in Hypersignal

Power Spectra Generation in Hypersignal

Transfer Function Generation in Hypersignal

Inverse FFT Generation in Hypersignal

Difference Equations in Hypersignal

Sampling Rate Conversion in Hypersignal

Hypersignal-Macro EX Functions

Long FFT in Hypersignal

Long Convolution/Correlation in Hypersignal

Frequency Zoom in Hypersignal

Minimum Phase in Hypersignal

Wavelet Transform in Hypersignal

Hypersignal-Acoustic Functions

Schroeder Integration in Hypersignal

Equalization in Hypersignal

MLS Stimulus & Response in Hypersignal

Schroeder Integration in Hypersignal

[@k integration,, Schroeder]

The schroed.mac program is included with Hypersignal-Acoustic software only.

When operating Hypersignal in user mode, the schroed.mac program allows Schroeder Integration to appear as a virtual function when invoked from the Macro Manager menu. The schroed.mac program prompts for an input and output time domain waveform filenames. Time domain input data can be in either .TIM or .WAV formats. The function performs a reverse integration of the input waveform, creating a monotonically decreasing representation of the input waveform's "potential energy", or remaining energy, with respect to time. Display of Schroeder integration results should be made using the Waveform Display/Edit function, in either linear or log amplitude mode. By calculating a line-fit to the slope between time points in log mode of this display, reverberation-time measurements can be estimated. For example, the time required for energy to drop 60 dB along a reasonable line-fit could be used as an estimate for an RT60 measurement value.

Additional Hypersignal Functions

FFT Generation in Hypersignal

Convolution/Correlation in Hypersignal
LPC Autocorrelation in Hypersignal
Recursive Filtering in Hypersignal
Power Spectra Generation in Hypersignal
Transfer Function Generation in Hypersignal
Inverse FFT Generation in Hypersignal
Difference Equations in Hypersignal
Sampling Rate Conversion in Hypersignal

Hypersignal-Macro EX Functions

Long FFT in Hypersignal
Long Convolution/Correlation in Hypersignal
Frequency Zoom in Hypersignal
Minimum Phase in Hypersignal
Wavelet Transform in Hypersignal

Hypersignal-Acoustic Functions

Schroeder Integration in Hypersignal
Equalization in Hypersignal
MLS Stimulus & Response in Hypersignal

Equalization in Hypersignal

The equalize.mac program is included only with Hypersignal-Acoustic software.

When operating Hypersignal in user mode, the equalize.mac program allows equalization to appear as a virtual function when invoked from the Macro Manager menu. The equalize.mac program takes Spectrum Analyzer transfer function display (either analog signal or waveform file as input) and writes inverted linear magnitude output to a .FRQ file. This is done using the Spectrum Analyzer 'X' (inverted write) command, which is active only in Hypersignal-Acoustic software. The

"zero dB" point used for inversion is the zero dB point on the transfer function display as default, or the current vertical position of the cursor, if displayed. The macro function then takes the inverted magnitude output, and the original phase file (which must be created beforehand if the input was analog), and calculates real/imag components. It then calculates an inverse FFT to create a time domain impulse response which can be used as an FIR filter to equalize the original transfer function curve. As a sanity check, the macro function calculates the magnitude response of this impulse response, and displays it, along with the original, non-inverted magnitude response, in the Magnitude Display function, using the overlay mode for comparison and a display range of ±70 dB.

It is important to note that the length of the resulting FIR filter will be equal to the FFT size in the transfer function display, regardless of the size of the time domain (measurement) window in the transfer function display. Reasonable curve-matching may occur only with FFT sizes of 512 points or more, which means that it is unlikely that supported DSP hardware can implement filters of this size in real-time inside Hypersignal, since all Hypersignal instrumentation functions implement FIR filters using standard multiply-accumulate time domain techniques. For these reasons, you will probably need the Real-Time Convolver to apply the equalization filters in real-time, using supported DSP/analog I/O hardware.

When running Hypersignal-Acoustic under Windows, it is possible to set up and control two separate DSP/analog I/O boards at once, using two copies of HS-Acoustic. The main system would control the measurement/instrumentation board, and the second copy the Real-Time Convolver board. In this way, the real-time Spectrum Analyzer can be used to monitor the effects of the equalization, and create new equalization filters, as needed.

After running equalize.mac, it may be desirable to run the resulting impulse response waveform (.tim file) through the minphase.mac program. The minimum phase function does not change the magnitude response, but it attempts to minimize the overall group delay. This may produce a more "normal-looking" impulse response, as well as shorten the number of taps required if the impulse response is to be downloaded and run as an FIR filter. However, phase preservation from the original, unequalized response will be destroyed.

The equalize.mac program has been extensively tested with frequency domain waveform files generated by the tefrd.exe and sysidrd.exe

utilities, which convert real/imag output data files from Techron's Sound Lab software, and Ariel's SYSid software, to Hypersignal .FRQ and .PHZ formats. The tefrd.exe and sysidrd.exe utilities can be conveniently controlled using the tefrd.mac and sysidrd.mac macro functions; these two utilities are included only with Hypersignal-Acoustic software.

When using a fixed-point DSP board at the same time as the equalize.mac function, it is suggested that the DSP ACCELERATOR field in the System Config menu be temporarily disabled. Even though the amount of error introduced by fixed-point FFT operations is small, it may be important in the equalization process, which is numerically sensitive.

Additional Hypersignal Functions

FFT Generation in Hypersignal

Convolution/Correlation in Hypersignal

LPC Autocorrelation in Hypersignal

Recursive Filtering in Hypersignal

Power Spectra Generation in Hypersignal

Transfer Function Generation in Hypersignal

Inverse FFT Generation in Hypersignal

Difference Equations in Hypersignal

Sampling Rate Conversion in Hypersignal

Hypersignal-Macro EX Functions

Long FFT in Hypersignal

Long Convolution/Correlation in Hypersignal

Frequency Zoom in Hypersignal

Minimum Phase in Hypersignal

Wavelet Transform in Hypersignal

Hypersignal-Acoustic Functions

Schroeder Integration in Hypersignal

Equalization in Hypersignal

MLS Stimulus & Response in Hypersignal

MLS Stimulus & Response in Hypersignal

[@k stimulus & response,, MLS, maximum length sequence]

The mls.mac program is included only with Hypersignal-Acoustic software. This function is still considered to be in a beta state, with improvements continuing to be added.

When operating Hypersignal in user mode, the mls.mac program allows MLS Stimulus & Response to appear as a virtual function when invoked from the Macro Manager menu. The mls.mac program prompts for stimulus length, number of averages, output waveform filename, sampling frequency, a channel list, and stimulus amplitude and offset. It uses the currently installed hardware specified in the System Configuration menu ANALOG CONVERSION field, performs the stimulus & response operation, synchronously averages the results in the time domain, performs a cross-correlation operation to extract the impulse response, and then an FFT to calculate the frequency response. The macro program concludes by showing the frequency response in the Magnitude Display, with vertical scale set to ±50 dB.

The mls.mac program calls the f_mls.mac, f_fft, and f_magdsp functions, which can also be called from an outside application when Hypersignal is being operated in engine mode.

Currently, f_mls.mac does not have an option to equalize out the response due solely to the measurement board being used. This option, and other "calibration mode" features, are expected to be added later.

MLS Stimulus & Response DSP Hardware Considerations

You must be aware of your DSP/analog I/O board memory size to make effective and reliable use of the mls.mac function. Some boards have a large amount of onboard DSP memory (either SRAM or

DRAM) and let you run huge MLS stimulus sizes, some boards have very little memory, and most are in-between. Because Hypersignal-Acoustic supports so many types of hardware, the mls.mac function is very general in nature. See DSPower Reference Guide section xx, MLS Stimulus & Response, for more information.

Below is a short list of example boards, the amount of data memory available for use by Hypersignal when shipped with Signalogic-recommended memory amounts, and the maximum stimulus size. Memory amounts are in words (natural wordlength of DSP device) and given in two values: Signalogic recommended amount that should be shipped with the board, and actual stimulus area available (excluding program area and input buffers). Memory amounts marked with an "*" can be packed double because the memory is accessible in words (16-bit values) as well as 32-bit values. Stimulus size values are in samples. See the Hypersignal-Macro Series Hardware Reference Guide for precise information about individual hardware types:

Table 6 MLS Stimulus Lengths vs. Hardware Type

Board	Total Mem	Stimulus Mem	Max Stimulus Size
Ariel DSP-32C	64k*	96k	32767
Ariel PC-56D+ ProPort	64k	20k	262143†
Ariel DSP-16	64k	20k	8191
CAC QUANTUMdsp	512k*	984k	262143
Innovative Integration PC31	128k	96k	32767
National Instruments AT-DSP2200	384k*	728k	262143
Signalogic Sig32C-8	128k*	192k	65535
Techron TEF20HI	48k	16k	262143†
TBS Tahiti	32k	12k	262143†

NOTE: †Indicates dynamic onchip generation of MLS output, which avoids stimulus memory limitation. Currently active only for some DSP5600x hardware and some stimulus lengths.

For larger stimulus sizes, it is highly recommended that DSP hardware be obtained with at least an initial memory expansion option, and possibly the maximum amount of memory expansion allowable. In many cases, the DSP hardware manufacturer does not ship boards with much beyond 16k words of memory, for two reasons: in order to keep cost down, and because traditionally DSP programs have been known for compactness. Thus, it is important to know what is the memory amount status of your hardware.

Another important hardware consideration to be aware of when using mls.mac is memory speed. Memory that runs with wait-states can severely affect the maximum sampling rate possible when using the mls.mac function. For example, boards that provide DRAM for data buffers rather than SRAM can require as many as 4-6 wait-states to access the DRAM. DRAM is used to provide huge amounts of memory for reasonable cost, but there is always a tradeoff. Another example is the inexpensive multimedia class of hardware, which typically use slower SRAM (about 3 wait-states) in order to keep costs down. Generally, use of slow SRAM will cut maximum MLS sampling rates to about 25 kHz, and use of DRAM will cut maximum rates to about 16 kHz to 22 kHz. However, degradation depends also on the processor type and speed, which can mitigate decreases in maximum rates.

Additional Hypersignal Functions

FFT Generation in Hypersignal

Convolution/Correlation in Hypersignal

LPC Autocorrelation in Hypersignal

Recursive Filtering in Hypersignal

Power Spectra Generation in Hypersignal

Transfer Function Generation in Hypersignal

Inverse FFT Generation in Hypersignal

Difference Equations in Hypersignal

Sampling Rate Conversion in Hypersignal

Hypersignal-Macro EX Functions

Long FFT in Hypersignal

Long Convolution/Correlation in Hypersignal

Frequency Zoom in Hypersignal

Minimum Phase in Hypersignal

Wavelet Transform in Hypersignal

Hypersignal-Acoustic Functions

Schroeder Integration in Hypersignal

Equalization in Hypersignal

MLS Stimulus & Response in Hypersignal

Hypersignal Filter Design Overview

[@k filter design,, overview]

All Hypersignal-Macro series software packages contain reasonable digital filter design, offering from 75% to 85% of the capability of specialized filter design packages. Where Hypersignal-Macro series software performs better is in the actual use of filters--either in waveform file-based simulation, or in real-time. In addition to basic DSP simulation functions such as convolution and recursive filtering, a number of functions and displays are available that can utilize FIR and IIR filter coefficient files as input, including real-time instruments and functions such as sampling rate conversion. Filter coefficient files can be created in two ways:

1. Using Hypersignal-Macro series software built-in FIR Filter Design and IIR Filter Design functions, which offer both classical and arbitrary design methods.
2. Creating the filter coefficient files outside of Hypersignal, based on user-defined filter design methods or specifications.

Hypersignal Digital Filter Functions

Hypersignal Filter Design Overview

FIR Filter Design in Hypersignal
IIR Filter Design in Hypersignal
Filter DSP Code Generation in Hypersignal
Pole-Zero IIR Filter Display in Hypersignal
File-Based Filter Simulation in Hypersignal
"Snap-In" Real-Time Filtering in Hypersignal

FIR Filter Design in Hypersignal

[@k filter design,, Hypersignal]

All Hypersignal-Macro series software packages contain built-in Kaiser Window and Parks-McClellan design of lowpass, highpass, bandpass, and bandstop FIR filters. Differentiation and Hilbert Transform designs, which are a subset of Parks-McClellan design, may be specified. Also, arbitrary designs, such as multiband and irregular or log frequency passbands, can be specified and realized using the Parks-McClellan method. Selectable filter name, sampling frequency, center frequency, bandwidth, transition bandwidths, stopband attenuation, and passband ripple. Filter length is estimated by the software, but may be controlled explicitly. All designs are first displayed and can be measured in a graphical filter display prior to design; arbitrary designs can be further edited directly using the mouse. Resulting frequency responses can be compared in the graphical display to original specifications and "tolerance regions".

Hypersignal filter design functions create as output a time domain impulse response waveform, and both an ASCII "filter coefficient file", which contains a numerator (pole) section and a denominator (zero) section (which is not used for FIR filters), and an ASCII "design goal file", which contains the filter specifications and the arbitrary frequency response, if any. Both of these latter two files can be created and manipulated directly, outside of Hypersignal, to achieve user-defined filters and numerically-specified design goals.

Current maximum FIR filter sizes are 4096 for Kaiser Window design, and 768 for Parks-McClellan design.

Hypersignal Digital Filter Functions

Hypersignal Filter Design Overview
FIR Filter Design in Hypersignal
IIR Filter Design in Hypersignal
Filter DSP Code Generation in Hypersignal
Pole-Zero IIR Filter Display in Hypersignal
File-Based Filter Simulation in Hypersignal
"Snap-In" Real-Time Filtering in Hypersignal

IIR Filter Design in Hypersignal

[@k filter design,, IIR]

All Hypersignal-Macro series software packages contain Butterworth, Chebyshev I and II, Bessel, and elliptic design of lowpass, highpass, bandpass, and bandstop IIR filters. Also, arbitrary designs, such as multiband and irregular or log frequency passbands, can be specified and realized using the Steiglitz method. Selectable filter name, sampling frequency, center frequency, bandwidth, transition bandwidths, stopband attenuation, and passband ripple or corner attenuation. In the case of Steiglitz design, the filter order and required minimum convergence error are also selectable. All designs are first displayed and can be measured in a graphical filter display prior to design; arbitrary designs can be further edited directly using the mouse. Resulting frequency responses can be compared in the graphical display to original specifications and "tolerance regions". Current maximum filter orders are 20 for Bessel designs, and 40 for Butterworth, Chebyshev I and II, elliptic and Steiglitz designs.

Hypersignal filter design functions create as output a time domain impulse response waveform, and both an ASCII "filter coefficient file", which contains a numerator (pole) section and a denominator (zero) section, and an ASCII "design goal file", which contains the filter specifications and the arbitrary frequency response, if any. Both of these files can be created and manipulated directly, outside of Hypersignal, to achieve user-defined filters and numerically-specified design goals.

Current maximum IIR filter order is 40 for bandpass or bandstop, and 20 for lowpass or highpass.

Hypersignal Digital Filter Functions

Hypersignal Filter Design Overview

FIR Filter Design in Hypersignal

IIR Filter Design in Hypersignal

Filter DSP Code Generation in Hypersignal

Pole-Zero IIR Filter Display in Hypersignal

File-Based Filter Simulation in Hypersignal

"Snap-In" Real-Time Filtering in Hypersignal

Filter DSP Code Generation in Hypersignal

[@k code generation,, filters]

All Hypersignal-Macro series software packages contain assembly language and C source code generation for FIR and IIR filters. Both floating-point and fixed point DSP devices are supported, including the Analog Devices ADSP-21xx, AT&T DSP32C and DSP3210, the Motorola DSP5600x and DSP96002, the NEC 77220, and the Texas Instruments TMS320C25, and TMS320C3x and TMS320C4x. For fixed-point DSP chips, quantization of filter coefficients and delay storage elements can be selected, ranging from 8 bits to the maximum word length of the DSP chip. A chip-independent quantization mode is also available, allowing quantization up to 32 bits. For IIR filters, cascade structure can be selected to be Direct Form, Modified Direct Form, and Balanced Pole-Zero. The quantization and cascade structure options are initialized by the software to optimum settings, depending on the filter type. Generated assembly language code is highly optimized and in loop form; quantization-related instructions are only inserted when necessary.

Also, simulated quantized impulse responses can be generated when working with fixed-point DSP devices. The quantized impulse responses can be processed with FFT Generation and compared to original filter design frequency responses in either the Magnitude Display function or in the filter design function, where tolerance region display and crosshair cursor measurement can be utilized to determine the effects of quantization.

Hypersignal Digital Filter Functions

Hypersignal Filter Design Overview

FIR Filter Design in Hypersignal

IIR Filter Design in Hypersignal

Filter DSP Code Generation in Hypersignal

Pole-Zero IIR Filter Display in Hypersignal

File-Based Filter Simulation in Hypersignal

"Snap-In" Real-Time Filtering in Hypersignal

Pole-Zero IIR Filter Display in Hypersignal

[@k filter display,, pole-zero]

All Hypersignal-Macro series software packages contain pole-zero display and analysis for IIR filters, created either from the built-in Hypersignal IIR filter design methods, or specified in user-defined IIR filter coefficient files. User-defined filters must be specified in Z-Domain 2nd-order factored form, or "biquad cascade" form, in IIR filter coefficient files. Z-Plane display is available for direct display of the filter coefficient file values. An inverse bilinear transformation is performed to allow normalized display of the filter transfer function in the S-Plane.

Hypersignal Digital Filter Functions

Hypersignal Filter Design Overview

FIR Filter Design in Hypersignal

IIR Filter Design in Hypersignal

Filter DSP Code Generation in Hypersignal

Pole-Zero IIR Filter Display in Hypersignal

File-Based Filter Simulation in Hypersignal

"Snap-In" Real-Time Filtering in Hypersignal

File-Based Filter Simulation in Hypersignal

[@k filtering, filter simulation]

For all Hypersignal-Macro series software packages, IIR filters designed in Hypersignal or created outside Hypersignal can be simulated using the Recursive Filtering function. The ASCII filter coefficient files offer both a convenient access point for you to take the results of Hypersignal filter design outside of Hypersignal, and for you to cause other Hypersignal filter-related functions to use your externally generated coefficients for further processing.

For all Hypersignal-Macro series software packages, FIR filters designed in Hypersignal or created outside Hypersignal can be simulated using the Convolution function.

Hypersignal Digital Filter Functions

Hypersignal Filter Design Overview

FIR Filter Design in Hypersignal

IIR Filter Design in Hypersignal

Filter DSP Code Generation in Hypersignal

Pole-Zero IIR Filter Display in Hypersignal

File-Based Filter Simulation in Hypersignal

"Snap-In" Real-Time Filtering in Hypersignal

"Snap-In" Real-Time Filtering in Hypersignal

[@k real-time filtering, filtering,, real-time]

For real-time versions of Hypersignal-Macro series software (all except Hypersignal-Macro NRT), IIR and FIR filters designed in Hypersignal or created outside Hypersignal can be "snapped-in" to the instrumentation and disk record/output functions for immediate real-time operation. In the case of the instrumentation displays, up to two filters can be enabled independently on dual-trace analog data,

quantization can be selected, and cascade structure can be controlled from the ASCII filter coefficient files. Filtering takes place prior to all instrumentation calculations, such as transfer function and impulse response, allowing filters to become an intrinsic part of the measurement process, enabling selective frequency band analysis and/or delays. In the disk record/output and stimulus & response functions, one filter can be enabled on one or two channels; quantization can be selected and cascade structure can be controlled as with the instrumentation displays. In addition to the Code Generation function listed above, the routines used for real-time filtering, which include dynamic control over quantization, are available in all DSP Source Code Interfaces.

Hypersignal Digital Filter Functions

Hypersignal Filter Design Overview

FIR Filter Design in Hypersignal

IIR Filter Design in Hypersignal

Filter DSP Code Generation in Hypersignal

Pole-Zero IIR Filter Display in Hypersignal

File-Based Filter Simulation in Hypersignal

"Snap-In" Real-Time Filtering in Hypersignal

Hypersignal Interactive Display Overview

[@k display,, overview, interactive display,, overview]

When operated in user mode, Hypersignal-Macro series software packages offer several types of interactive displays which can be accessed directly.

When Hypersignal is opeating in engine mode, interactive displays can be accessed indirectly, in which case the user-interface is provided by the controlling application, and display and/or status results are sent to the engine via the Pipe.386 VxD (see section 1.1, Relationship of Signalogic DSP Software Components, for more information).

Interactive controls typically include zoom, translate, vertical scale, cursor readout, target window measurements, store to file, printout, frequency axis control, and more.

In the time domain Waveform Display/Edit display, rubberband and point-mode editing of .TIM and .WAV waveform files is active.

Also, in the case of real-time versions of Hypersignal-Macro series software, D/A playback is available from the waveform display/edit and joint time-frequency contour displays.

Hypersignal Display Functions

Hypersignal Interactive Display Overview

Time Domain Waveform Display/Edit in Hypersignal

Frequency Domain Magnitude and Phase Display in Hypersignal

Joint Time-Frequency Waterfall in Hypersignal

Joint Time-Frequency Contour in Hypersignal

Filter Design Display in Hypersignal

Pole-Zero Display in Hypersignal

Time Domain Waveform Display/Edit in Hypersignal

[@k display,, time domain, display,, .WAV, .WAV display]

All Hypersignal-Macro series software packages contain a time domain Waveform Display/Edit function which can be accessed directly when operating in user mode, and indirectly when Hypersignal is in engine mode. In the latter case, the user-interface is provided by the controlling application, and display results are sent to the engine via the Pipe.386 VxD (see section 1.1, Relationship of Signalogic DSP Software Components, for more information).

The Waveform Display/Edit function is a general display that allows single-, dual-, or overlaid-trace display and editing of one or two waveforms. In the case of a single waveform, the data can be two-channel or complex. Also, waveforms other than time domain data can be displayed and edited. Both cut-and-paste and free-form "rubber

band" editing modes are available. These modes can be freely switched between in editing sessions, and are fully enabled at all levels of zoom. Cut-and-paste operations include delete, copy, insert, and overwrite. Display controls include "current active trace" selection, framesize, zoom (and zoom-out), pan, amplitude scaling, random-access, target window and crosshair cursor measurement and readout, units, color and background, grid, and selection of vertical axis units, scaling factor, and maximum value.

In Hypersignal-Acoustic software, a log amplitude mode of display is also available, which depicts data as $10\log_{10}[x^2]$. All display and editing controls are active in this mode, which can easily be toggled on and off as needed.

Additionally, segments of displayed waveforms can be written to file or sent to the D/A output of the currently selected DSP/acquisition hardware. A built-in print function is also available that allows screen dumps or publication-quality EMS memory regenerations to nearly 200 different printers.

Hypersignal Display Functions

Hypersignal Interactive Display Overview

Time Domain Waveform Display/Edit in Hypersignal

Frequency Domain Magnitude and Phase Display in Hypersignal

Joint Time-Frequency Waterfall in Hypersignal

Joint Time-Frequency Contour in Hypersignal

Filter Design Display in Hypersignal

Pole-Zero Display in Hypersignal

Frequency Domain Magnitude and Phase Display in Hypersignal

[@k display,, magnitude, display,, phase, magnitude display, phase display]

All Hypersignal-Macro series software packages contain frequency domain Magnitude and Phase Display functions which can be accessed directly when operating in user mode, and indirectly when

Hypersignal is in engine mode. In the latter case, the user-interface is provided by the controlling application, and display results are sent to the engine via the Pipe.386 VxD (see section 1.1, Relationship of Signalogic DSP Software Components, for more information).

The Magnitude Display function allows frame-by-frame display and inspection of frequency domain magnitude data. The Phase Display function operates similarly, but for frequency domain phase data, and includes a group delay option and a "phase unwrapping" option, much like a Bode plot. These functions allow single-, dual-, or overlaid-trace display of one or two frequency domain waveforms.

For Magnitude Display, controls include log or linear amplitude or amplitude-squared display, "current active trace" selection, zoom (and zoom-out), pan, vertical amplitude limits, amplitude scaling, random-access, target window and crosshair cursor measurement and readout, units, color and background, grid, and selection of vertical axis units, scaling factor, and maximum value. For Phase Display, controls include wrapped phase, unwrapped phase, or group delay display format, "current active trace" selection, zoom (and zoom-out), pan, amplitude scaling, random-access, target window and crosshair cursor measurement and readout, units, color and background, grid, and selection of group delay smoothing factor.

For both types of display, the frequency axis can be specified to be linear or log. In the log mode, the log base, number of cycles to display, and minimum frequency axis value can be selected.

A built-in print function is also available that allows screen dumps or publication-quality EMS memory regenerations to nearly 200 different printers.

Hypersignal Display Functions

Hypersignal Interactive Display Overview

Time Domain Waveform Display/Edit in Hypersignal

Frequency Domain Magnitude and Phase Display in Hypersignal

Joint Time-Frequency Waterfall in Hypersignal

Joint Time-Frequency Contour in Hypersignal

Filter Design Display in Hypersignal

Pole-Zero Display in Hypersignal

Joint Time-Frequency Waterfall Display in Hypersignal

[@k frequency domain display, waterfall display]

All Hypersignal-Macro series software packages contain a frequency domain joint time-frequency waterfall display function which can be accessed directly when operating in user mode, and indirectly when Hypersignal is in engine mode. In the latter case, the user-interface is provided by the controlling application, and display results are sent to the engine via the Pipe.386 VxD (see section 1.1, Relationship of Signalogic DSP Software Components, for more information).

The waterfall version of joint time-frequency display is sometimes referred to as a "landscape" or "spectrogram" display, and is labeled on the Hypersignal main menu as the 3-D Spectrogram. In a 3-D Spectrogram display, multiple FFT output frames are displayed at once, with frequency along the horizontal axis, amplitude along the vertical, and time depicted as an "apparent", or z-axis going into the screen. Each successive FFT output frame is displayed behind the previous and hidden-line removal may be enabled, giving the overall appearance of a three-dimensional display. (Note: when changes between successive frames are relatively minor, the overall appearance of the display is similar to a cascading waterfall display.)

The 3-D Spectrogram allows single- or dual-trace display of one or two waveforms. In the case of a single waveform, the data can be two-channel. Controls include amount of data shown, log or linear amplitude or amplitude-squared display, vertical amplitude limits, zoom (and zoom-out), pan, random-access, target window measurement and readout, units, color and background, grid, angle of apparent incline (ranging from 5 to 90), and hidden-line removal and surface mesh options. Also, the frequency axis can be specified to be linear or log. In the log mode, the log base, number of cycles to display, and minimum frequency axis value can be elected.

A built-in print function is also available that allows screen dumps or publication-quality EMS memory regenerations to nearly 200 different printers.

Hypersignal Display Functions

Hypersignal Interactive Display Overview

Time Domain Waveform Display/Edit in Hypersignal
Frequency Domain Magnitude and Phase Display in Hypersignal
Joint Time-Frequency Waterfall in Hypersignal
Joint Time-Frequency Contour in Hypersignal
Filter Design Display in Hypersignal
Pole-Zero Display in Hypersignal

Joint Time-Frequency Contour Display in Hypersignal

[@k frequency domain display, contour display]

All Hypersignal-Macro series software packages contain a frequency domain joint time-frequency contour display function which can be accessed directly when operating in user mode, and indirectly when Hypersignal is in engine mode. In the latter case, the user-interface is provided by the controlling application, and display results are sent to the engine via the Pipe.386 VxD (see section 1.1, Relationship of Signalogic DSP Software Components, for more information).

This display is sometimes referred to as a "spectrograph", and is labeled on the Hypersignal main menu as the 2-D Spectrograph. In a 2-D Spectrograph display, multiple FFT output frames are displayed at once, with frequency along the vertical axis, time along the horizontal, and linear or log amplitude encoded either with different colors or gray shades. In this format, you are "looking down" at the frequency domain data--for this reason, the 2-D Spectrograph display is often referred to as a form of "contour plot." This type of display allows more joint time-frequency information to be displayed and to be viewable at one time than any other.

The 2-D Spectrograph allows single-, or dual-trace display of one or two waveforms. In the case of a single waveform, the data can be two-channel. Controls include "current active trace" selection, amount of data shown, vertical amplitude limits (which effectively control the amount of quantized amplitude in linear mode or the amount of quantized dB in log mode that is assigned to each color or gray shade), zoom (and zoom-out), pan, random-access, target window and crosshair cursor measurement and readout, units, color and background, and surface topology. Also, the frequency axis can be

specified to be linear or log. In log mode, the log base, number of cycles to display, and minimum frequency axis value can be selected.

Additionally, segments of displayed waveforms can be sent to the D/A output of the currently selected DSP/acquisition hardware. A built-in print function is also available that allows screen dumps or publication-quality EMS memory regenerations to nearly 200 different printers. Some color printers are supported. For black & white printers, gray level appearance is achieved through dot-density control.

Hypersignal Display Functions

Hypersignal Interactive Display Overview

Time Domain Waveform Display/Edit in Hypersignal

Frequency Domain Magnitude and Phase Display in Hypersignal

Joint Time-Frequency Waterfall in Hypersignal

Joint Time-Frequency Contour in Hypersignal

Filter Design Display in Hypersignal

Pole-Zero Display in Hypersignal

Filter Design Display in Hypersignal

[@k display,, filters]

All Hypersignal-Macro series software packages contain a graphical filter design and measurement display which can be accessed directly when operating in user mode, and indirectly when Hypersignal is in engine mode. In the latter case, the user-interface is provided by the controlling application, and display results are sent to the engine via the Pipe.386 VxD (see section 1.1, Relationship of Signalogic DSP Software Components, for more information).

Proposed FIR and IIR filter designs are always displayed graphically in Hypersignal prior to actual filter construction. Several measurement functions are available, which apply to both classical and arbitrary designs. Several editing functions are also available, which apply only to arbitrary designs. Measurement functions include target window and crosshair measurement and readout, grid, log/linear frequency and amplitude axes, overlay of actual and desired frequency

responses, vertical scaling, and tolerance region display toggle. Editing functions include cut-and-paste and free-form rubberband modes, control over frequency point-density and "defined" and "undefined" regions, and initialization of response from menu specifications or design goal files. More information about filter design display is available under the Filter Design section, above.

Hypersignal Display Functions

Hypersignal Interactive Display Overview

Time Domain Waveform Display/Edit in Hypersignal

Frequency Domain Magnitude and Phase Display in Hypersignal

Joint Time-Frequency Waterfall in Hypersignal

Joint Time-Frequency Contour in Hypersignal

Filter Design Display in Hypersignal

Pole-Zero Display in Hypersignal

Pole-Zero Display in Hypersignal

[@k display,, pole-zero]

All Hypersignal-Macro series software packages contain a pole-zero analysis display which can be accessed directly when operating in user mode, and indirectly when Hypersignal is in engine mode. In the latter case, the user-interface is provided by the controlling application, and display results are sent to the engine via the Pipe.386 VxD (see section 1.1, Relationship of Signalogic DSP Software Components, for more information).

The Pole-Zero Display function provides display and analysis for IIR filters, created either from the built-in Hypersignal IIR filter design methods, or specified in user-defined IIR filter coefficient files. User-defined filters must be specified in Z-Domain 2nd-order factored form, or "biquad cascade" form, in IIR filter coefficient files. Z-Plane display is available for direct display of the filter coefficient file values. An inverse bilinear transformation is performed to allow normalized display of the filter transfer function in the S-Plane.

Hypersignal Display Functions

Hypersignal Interactive Display Overview

Time Domain Waveform Display/Edit in Hypersignal

Frequency Domain Magnitude and Phase Display in Hypersignal

Joint Time-Frequency Waterfall in Hypersignal

Joint Time-Frequency Contour in Hypersignal

Filter Design Display in Hypersignal

Pole-Zero Display in Hypersignal

Hypersignal Instrumentation & Measurement Overview

[@k overview,, instrumentation & measurement]

Real-time Hypersignal-Macro series software packages (all except Hypersignal-Macro NRT) contain several instrumentation and measurement functions which work in conjunction with supported DSP/analog hardware.

When Hypersignal is operating in user mode, instrument functions can be accessed directly. When Hypersignal is opeating in engine mode, instrument displays can be accessed indirectly, in which case the user-interface is provided by the controlling application, and display and/or status results are sent to the engine via the Pipe.386 VxD (see section 1.1, Relationship of Signalogic DSP Software Components, for more information).

Instrument functions typically download DSP executable files to the supported hardware, and then present interactive controls and/or displays that allow operation of the DSP code to be changed dynamically. Executable files for several supported DSP chip families are included with Hypersignal software; customization of these files or creation of new ones is possible. See section 1.2.3, DSP Source Code Interface Software Summary, for more information.

Hypersignal Instrumentation & Measurement Functions

Hypersignal Instrumentation & Measurement Overview
Spectrum Analyzer in Hypersignal
Digital Oscilloscope in Hypersignal
Continuous Record/Output To/From Disk in Hypersignal
Stimulus & Response in Hypersignal
Signal Synthesis in Hypersignal

Spectrum Analyzer in Hypersignal

[@k instrumentation,, spectrum analyzer]

All real-time Hypersignal-Macro series software packages contain a version of a Spectrum Analyzer instrument function which can be accessed directly when operating in user mode, and indirectly when Hypersignal is in engine mode. In the latter case, the user-interface is provided by the controlling application, and display results are sent to the engine via the Pipe.386 VxD (see section 1.1, Relationship of Signalogic DSP Software Components, for more information).

The Hypersignal Spectrum Analyzer can display single-, dual-, or overlaid-trace input from previously stored waveform files or any of the currently active analog channel inputs. Various combinations are possible, including viewing the same analog channel in different domains, formats, or with different filtering options, and overlaying analog input with previously stored waveform file input. Selectable display data formats include time, linear/log magnitude, phase, transfer function, and impulse response (iFFT of transfer function). Averaging can be enabled. Operation from the DSP chip point of view is always in real-time up to the maximum sampling rate, but the rate of update of display frames to the video system of the host computer depends on the speed of the host I/O bus and video system. The maximum sampling rate possible depends on the DSP/acquisition board and DSP chip in use.

FIR and IIR filters created using Hypersignal filter design or supplied from outside of Hypersignal can be "snapped-in" to each trace independently and applied to analog input data in real-time (see "Snap-In" Real-Time Filtering, above). Maximum filter sizes and real-time operation with filters are dependent on the sampling rate and DSP/acquisition board in use.

Controls include timebase, trigger level, trace, and mode, vertical scaling and position, units, color and background, and crosshair cursor measurement, readout, and calibration (in engineering dB units). Trace displays can be captured to waveform file, or sent to one of about 200 printers as a screen dump or publication-quality EMS memory regeneration. Fourier parameters such as FFT size, time domain framesize, and time domain window can be specified. Also, the software supports a "3-D map mode", which creates a waterfall type of display, in order to allow a time-history aspect in frequency data format display.

Frequency axes can be linear or log. In the log mode, the log base, number of cycles to display, and minimum frequency axis value can be selected. Also, 1/N octave band display can be enabled, which causes a "bar-graph" type of display.

For Hypersignal-Acoustic software only, trace displays can be inverted and stored to an FIR filter design goal file, and to magnitude and phase waveform files. An FIR filter can then be designed that approximates the inverted trace and applied in real-time to equalize the frequency response. Also for Hypersignal-Acoustic software only, real and imaginary transfer function data can be captured to waveform file, and a coherence display format is currently being installed.

When possible, the hardware resources of the DSP/acquisition board are under direct control. The sampling rate, analog domain anti-alias filtering (if programmable), input source, and hardware gain (if programmable) can be controlled from the software. DSP/acquisition hardware supported by Hypersignal offer varied functionality, the best of which include maximum sampling rates of 10 MHz, up to 64 channels, programmable anti-alias filters, programmable input gain from 0 to 60 dB, and programmable input source selection ranging from line-level to microphone to AC power line.

Hypersignal Instrumentation & Measurement Functions

Hypersignal Instrumentation & Measurement Overview

Spectrum Analyzer in Hypersignal

Digital Oscilloscope in Hypersignal

Continuous Record/Output To/From Disk in Hypersignal

Digital Oscilloscope in Hypersignal

[@k instrumentation,, digital oscilloscope]

All real-time Hypersignal-Macro series software packages contain a version of a Digital Oscilloscope instrument function which can be accessed directly when operating in user mode, and indirectly when Hypersignal is in engine mode. In the latter case, the user-interface is provided by the controlling application, and display results are sent to the engine via the Pipe.386 VxD (see section 1.1, Relationship of Signalogic DSP Software Components, for more information).

The Hypersignal Digital Oscilloscope can display single-, dual-, or overlaid-trace time domain input from previously stored waveform files or any of the currently active analog channel inputs. Various combinations are possible, including viewing the same analog channel with different filtering options. Operation from the DSP chip point of view is always in real-time up to the maximum sampling rate, but the rate of update of display frames to the video system of the host computer depends on the speed of the host I/O bus and video system. The maximum sampling rate possible depends on the DSP/acquisition board and DSP chip in use.

FIR and IIR filters created using Hypersignal filter design or supplied from outside of Hypersignal can be "snapped-in" to each trace independently and applied to analog input data in real-time (see "Snap-In" Real-Time Filtering, above). Maximum filter sizes and real-time operation with filters are dependent on the sampling rate and DSP/acquisition board in use.

Controls include timebase, trigger level, trace, and mode, vertical scaling and position, units, and color and background. Trace displays can be captured to waveform file, or sent to one of about 200 printers as a screen dump or publication-quality EMS memory regeneration.

When possible, the hardware resources of the DSP/acquisition board are under direct control. The sampling rate, analog domain anti-alias filtering (if programmable), input source, and hardware gain (if programmable) can be controlled from the software. DSP/acquisition hardware supported by Hypersignal offer varied functionality, the best

of which include maximum sampling rates of 10 MHz, up to 64 channels, programmable anti-alias filters, programmable input gain from 0 to 60 dB, and programmable input source selection ranging from line-level to microphone to AC power line.

Hypersignal Instrumentation & Measurement Functions

Hypersignal Instrumentation & Measurement Overview

Spectrum Analyzer in Hypersignal

Digital Oscilloscope in Hypersignal

Continuous Record/Output To/From Disk in Hypersignal

Stimulus & Response in Hypersignal

Signal Synthesis in Hypersignal

Continuous Record/Output To/From Disk in Hypersignal

[@k record to file, record to disk, file record, disk record, data acquisition, acquisition,, waveform file, waveform files,, recording, files,, recording]

All real-time Hypersignal-Macro series software packages contain an Analog Conversion instrument function which can be accessed directly when operating in user mode, and indirectly when Hypersignal is in engine mode. In the latter case, the user-interface is provided by the controlling application, and status and some data-monitoring results are sent to the engine via the Pipe.386 VxD (see section 1.1, Relationship of Signalogic DSP Software Components, for more information).

The Analog Convesion functions can perform continuous input and output of multichannel data to and from waveform files. Controls include sampling rate, hardware gain (if programmable), triggering (channel and level), software scaling and offsets, and amount of data. The maximum sampling rate possible depends on the number of channels in use, the selected DSP/acquisition hardware, and the speed of the host computer system (especially disk and I/O bus).

FIR and IIR filters created using Hypersignal filter design or supplied from outside of Hypersignal can be "snapped-in" and applied to analog input and output data in real-time (see "Snap-In" Real-Time Filtering, above). One real-time filter can be enabled for the input or output; for two-channel operation, the filter applies to both channels. Maximum filter sizes and real-time operation with filters are dependent on the sampling rate and DSP/acquisition board in use.

When possible, the hardware resources of the DSP/acquisition board are under direct control. The sampling rate, analog domain anti-alias filtering (if programmable), input source, and hardware gain (if programmable) can be controlled from the software.

Hypersignal Instrumentation & Measurement Functions

Hypersignal Instrumentation & Measurement Overview

Spectrum Analyzer in Hypersignal

Digital Oscilloscope in Hypersignal

Continuous Record/Output To/From Disk in Hypersignal

Stimulus & Response in Hypersignal

Signal Synthesis in Hypersignal

Stimulus & Response in Hypersignal

[@k instrumentation, stimulus & response]

All real-time Hypersignal-Macro series software packages contain an Analog Conversion function capable of analog simultaneous input which can be used for stimulus & response measurement. The Analog Conversion function can be accessed directly when operating in user mode, and indirectly when Hypersignal is in engine mode. In the latter case, the user-interface is provided by the controlling application, and status and some data-monitoring results are sent to the engine via the Pipe.386 VxD (see section 1.1, Relationship of Signalogic DSP Software Components, for more information).

A single-channel or two-channel stimulus & response measurement function is available that allows simultaneous output of stimuli waveform and continuous recording of analog input to disk. Controls

include sampling rate, hardware gain (if programmable), triggering (channel and level), software scaling and offsets, amount of data, and positive or negative stimulus delay. A positive delay directs recording to start after the onset of the stimulus; a negative delay directs recording to start before the stimulus. Operation from both the DSP chip and host computer point of view is always in real-time up to the maximum sampling rate. The maximum sampling rate possible depends on the number of channels in use, the selected DSP/acquisition hardware, and the speed of the host computer system (especially the disk and I/O bus).

FIR and IIR filters created using Hypersignal filter design or supplied from outside of Hypersignal can be "snapped-in" and applied to analog input data in real-time (see "Snap-In" Real-Time Filtering, above). One real-time filter can be enabled for the input only; for two-channel operation, the filter applies to both channels. Maximum filter sizes and real-time operation with filters are dependent on the sampling rate and DSP/acquisition board in use.

In Hypersignal-Acoustic software, a single-step MLS Stimulus & Response macro program (mls.mac) is available that generates a pseudo-random maximum length sequence (MLS) stimulus, outputs the stimulus while continuously recording the response of the system under test, and then performs cross-correlation and FFT operations to calculate the impulse and frequency responses of the system. The mls.mac program offers averaging, amplitude and offset control, and other features.

See Also

MLS Stimulus & Response in Hypersignal

Hypersignal Instrumentation & Measurement Functions

Hypersignal Instrumentation & Measurement Overview

Spectrum Analyzer in Hypersignal

Digital Oscilloscope in Hypersignal

Continuous Record/Output To/From Disk in Hypersignal

Stimulus & Response in Hypersignal

Signal Synthesis in Hypersignal

[@k continuous signal generation, synthesis,, signal]

All real-time Hypersignal-Macro series software packages contain an Analog Conversion function capable of continuous, repetitive analog output which can be used for signal generation. The Analog Conversion function can be accessed directly when operating in user mode, and indirectly when Hypersignal is in engine mode. In the latter case, the user-interface is provided by the controlling application, and status and some data-monitoring results are sent to the engine via the Pipe.386 VxD (see section 1.1, Relationship of Signalogic DSP Software Components, for more information).

Continuous signal synthesis in Hypersignal is implemented by output looping in the Analog Conversion function. By creating short signal/function waveforms with the Difference Equations function, and then using these waveforms as input to the Analog Conversion function, a continuous signal generator can be achieved when the ACQ/GEN field is set to 'GL'. Some guidelines to follow when specifying a Difference Equation:

- waveform lengths (as specified in the Difference Equations # OF SAMPLES field) should be an integer multiple of the function period; if you are in doubt about the function period, use the Waveform Display/Edit option to measure it exactly
- the overall waveform length should be anywhere from a few hundred samples to less than 6000 samples; only one or just a few periods of the function waveform is not recommended, because of the overhead required by the DSP software when "wrapping around" to repeat the function. And, definitely, long waveforms are not recommended because they become split into buffer segments due to DSP board memory limitations, defeating the continuous mechanism altogether
- the length of the waveform, in samples (and multiplied by the number of channels), should be divisible by two
- watch out for sampling frequency when designing the difference equation; most hardware only allows certain sampling frequencies, and if you base your calculations on a

sampling frequency the hardware cannot achieve, then there is likely to be a discontinuity when the output waveform repeats

Hypersignal Instrumentation & Measurement Functions

Hypersignal Instrumentation & Measurement Overview

Spectrum Analyzer in Hypersignal

Digital Oscilloscope in Hypersignal

Continuous Record/Output To/From Disk in Hypersignal

Stimulus & Response in Hypersignal

Signal Synthesis in Hypersignal

Macro Language: Hypersignal Programmability

[@j Hypersignal macro language, @k programmability,, macro language]

The Hypersignal macro language can be described as a "watered-down combination of Pascal and C". As with other languages, variables and programming constructs are available to store results, isolate often-used segments of code, invoke standard mathematical, file, and I/O functions, repeat segments of code, and make decisions based on results--in general, to solve problems and allow "what-if" experimentation. Unlike other languages, the macro language provides a direct "hot-line" to Hypersignal functions and displays, not only allowing them to be invoked and combined in any sequence, but to accept macro language variables as input parameters. This allows the numerous functions and displays in Hypersignal to appear as a library of callable routines. Some overall features of the Hypersignal macro language:

1. Wide range of numerical and string operations, including 32-bit floating-point, 32-bit integer, and 40-character string operations.
2. If-then, for-loop, and nested procedure constructs.

3. Built-in file I/O which "knows" about .TIM and .WAV waveform file formats.

4. "Interactive commands" which can be issued inside displays and instruments.

5. Temporary or permanent transition to manual control from executing macro language programs.

6. Built-in Hypersignal macro language compiler allows macro language programs to be used and distributed in exectuable form; a "make mode" is available to avoid compilation of source code files that have not changed.

Following is information about important macro language features, includng partial lists of operators, commands, and functions, and also a source code listing example:

Related Macro Language Topics

Macro Language: Hypersignal Programmability

Macro Record Mode

Hypersignal Macro Programs

Creating Macro Language Programs and Functions

Macro Language Variables, Constructs, and Procedures

Macro Language Operators

Macro Language Built-In Functions

Macro Language Interactive Commands

Macro Language Source Code Example

Macro Record Mode

[@k recording,, macros]

For repetitive tasks, a "record mode" can be enabled, in which the software records manual function and display steps to a macro source file, which can be "played back" afterwards.

Related Macro Language Topics

Macro Language: Hypersignal Programmability
Macro Record Mode
Hypersignal Macro Programs
Creating Macro Language Programs and Functions
Macro Language Variables, Constructs, and Procedures
Macro Language Operators
Macro Language Built-In Functions
Macro Language Interactive Commands
Macro Language Source Code Example

Hypersignal Macro Programs

[@k macro language programs]

Several ready-to-execute macro programs are provided with Hypersignal software. To achieve manually in Hypersignal, these would require multiple steps, as well as error-prone and distracting "thinking": intermediate calculation and what function sequence to use. The standard Hypersignal-Macro package includes sampling rate conversion and several examples of operations such as cut-and-paste waveform editing, D/A output from displays, etc. The Hypersignal-Macro EX package includes more powerful macro programs, such as frequency zoom and single-step stimulus & response. Hypersignal-Acoustic software includes still more macro programs, such as Schroeder integration and utilities to read Ariel SYSid and Techron SoundLab output files.

See also Programs, Template Files, and Function Files in section 1.1, Relationship of Signalogic DSP Software Components.

Related Macro Language Topics

Macro Language: Hypersignal Programmability
Macro Record Mode
Hypersignal Macro Programs
Creating Macro Language Programs and Functions

Macro Language Variables, Constructs, and Procedures

Macro Language Operators

Macro Language Built-In Functions

Macro Language Interactive Commands

Macro Language Source Code Example

Creating Macro Programs and Functions

[@k function files,, macro language, programs,, macro language]

A macro language program can consist of one or more macro language source files. A common approach is to create a main macro language *program* source file, with a descriptive name to be used as the program name for user-interface purposes, and one or more macro language *function* source files which contain procedures. The function files are then included at compile-time using the macro language include-file construct. Many of the ready-to-execute macro language programs that come with Hypersignal-Macro EX and Hypersignal-Acoustic take this approach. DSPower series software, when using a Hypersignal-Macro series software package in engine mode, use this technique. See DSP Engines and Programs, Template Files, and Function Files in section 1.1, Relationship of Signalogic DSP Software Components.

Macro language source files can be created using traditional programming techniques (a text editor!), the built-in Hypersignal Macro Record Mode, or a combination of both. Source files are then compiled by the built-in Hypersignal-Macro compiler, which in turn creates a single "macro object file" representing the macro program. This object file is what Hypersignal actually executes when the program is run, when operating in either user mode or engine mode.

After compiling once, only the macro object file need be accessed to execute a macro program repeatedly. A "compiler mode" can be set in the Macro Manager menu that controls the circumstances under which compiling occurs. A "make" setting automatically compiles source files only if they have been changed.

Related Macro Language Topics

Macro Language: Hypersignal Programmability
Macro Record Mode
Hypersignal Macro Programs
Creating Macro Language Programs and Functions
Macro Language Variables, Constructs, and Procedures
Macro Language Operators
Macro Language Built-In Functions
Macro Language Interactive Commands
Macro Language Source Code Example

Macro Language Variables, Constructs, and Procedures

[@k variables,, macro language, procedures,, macro language]

As with most high-level languages, variables must be "declared" to be one of several types before use, procedures can be defined before the "main body" of source code, mathematical operators and functions can be invoked, and various conditional and looping constructs are available. However, the macro language approach also allows unique ability, such as a simplified means of invoking native Hypersignal-Macro series functions and displays, and graceful transitions between "interactive" and "automatic" modes. For example, the system can stop in the middle of an automated test procedure, wait for the operator to make measurements and perform operations, and then resume the test procedure.

Variables and constants can be 32-bit integer, 32-bit floating-point (real), or 40-character string. There is also a boolean pseudo-type which is essentially treated as an integer. Constants can be indicated in decimal, floating-point, or hexadecimal.

Related Macro Language Topics

Macro Language: Hypersignal Programmability
Macro Record Mode
Hypersignal Macro Programs

Creating Macro Language Programs and Functions
Macro Language Variables, Constructs, and Procedures
Macro Language Operators
Macro Language Built-In Functions
Macro Language Interactive Commands
Macro Language Source Code Example

Macro Language Operators

[@k operators,, macro language]

Below is a partial list of Hypersignal macro language operators:

Table 7 Partial List of Hypersignal Macro Language Operators

+	addition
–	subtraction
/	division
*	multiplication
div	integer division
|	bitwise OR
&	bitwise AND
^	bitwise XOR

Related Macro Language Topics

Macro Language: Hypersignal Programmability
Macro Record Mode
Hypersignal Macro Programs
Creating Macro Language Programs and Functions
Macro Language Variables, Constructs, and Procedures
Macro Language Operators
Macro Language Built-In Functions

Macro Language Interactive Commands

Macro Language Source Code Example

Macro Language Built-In Functions

[@k functions,, macro language]

Several built-in functions are available in the Hypersignal macro language. Moreover, the language contains "interactive commands" that allow waveform and instrument displays to be controlled from macros. Below is a partial list of macro language functions and interactive commands. "Ident" refers to an identifier: a constant or variable (real, integer, or string):

Table 8 Partial List of Macro Language Built-In Functions

Abort	Causes the currently executing macro to abort Hypersignal without saving the user environment configuration file.
Abs(var)	Returns |var|.
Delay(N)	Macro execution is delayed for N msec.
Display(ident)	Displays up to 40 characters of the resulting text in the System Status area.
DOS Shell	Invokes a DOS shell, which brings up a DOS command line and temporarily transfers control to the keyboard.
Exit	Causes the currently executing macro to exit Hypersignal after saving the user environment configuration file.
Exp(var)	Returns e**vare^{var}.
var = GetDataValue (filename, N)	Assigns the Nth data value

	(sample) from the waveform specified by "filename" to the variable "var".
var = GetDateTime()	Returns and assigns the current date and time in packed form to "var".
var = GetHeaderValue (filename, N)	Assigns the Nth value from the header of the waveform specified by "filename" to the variable "var".
var = GetUserSetup(N)	Assigns the string value of the Nth User Setup field to the variable "var".
Ln(var)	Returns ln(var).
Log(var)	Returns Log_{10}(var).
PutDataValue (filename, N, value)	Puts "value" into the Nth position of the data section of the waveform specified by "filename".
PutHeaderValue (filename, N, value)	Puts "value" into the Nth position of the header of the waveform specified by "filename".
Round(var)	Returns var rounded to nearest integer.
Sqr(var)	Returns var^2.
Sqrt(var)	Returns sqrt(var)√var.
TenX(var)	Returns 10**var10^{var}.
Trace(N)	Enables or disables interactive command trace display.
Trunc(var)	Returns var truncated to nearest integer.
WaitForDateTime(DT)	Stops macro execution until the system date and time

	reaches the value DT.
WaitForKey(K)	Stops macro execution until the key 'K' is pressed at the keyboard.

Related Macro Language Topics

Macro Language: Hypersignal Programmability

Macro Record Mode

Hypersignal Macro Programs

Creating Macro Language Programs and Functions

Macro Language Variables, Constructs, and Procedures

Macro Language Operators

Macro Language Built-In Functions

Macro Language Interactive Commands

Macro Language Source Code Example

Macro Language Interactive Commands

[@k interactive commands,, macro language]

Several "interactive commands" are available in the Hypersignal macro language. These commands allow waveform and instrument displays to be controlled from macros. In the partial list of interactive commands below, "Ident" refers to an identifier: either a constant or variable (real, integer, or string):

Table 9 Partial List of Macro Language Interactive Commands

$Del	Invokes cut-and-paste delete prompt.
$Delay(N)	Macro execution is delayed for N msec.
$Down	Makes the target window smaller, or moves the crosshair cursor down.
$Ins	Invokes cut-and-paste insert prompt.

$Left	Moves the target window or the crosshair cursor to the left.
$Manual	Stops macro execution and returns controls to manual mode. Execution of the macro program is halted until the current interactive display is exited (manually) by the user. All controls are active in manual mode.
$PanB	Pans backward in waveform displays.
$PanF	Pans forward in waveform displays.
$PgDn	Reduces amplitude scaling in waveform displays.
$PgUp	Increases amplitude scaling in waveform displays.
$Print	Invokes the pop-up print menu in waveform displays
$Right	Moves the target window or the crosshair cursor to the right.
$Trace(N)	Enables or disables interactive command trace display.
$Unzoom	Returns to previous level of zoom.
$Up	Makes the target window larger, or moves the crosshair cursor up
$WaitForDateTime(DT)	Stops macro execution until the system date and time reach the value of DT.
$WaitForKey(K)	Stops macro execution until the key 'K' is pressed at the keyboard.
$Zoom	In waveform displays, executes ZOOM interactive display command.

Related Macro Language Topics

Macro Language: Hypersignal Programmability

Macro Record Mode

Hypersignal Macro Programs

Creating Macro Language Programs and Functions

Macro Language Variables, Constructs, and Procedures

Macro Language Operators

Macro Language Built-In Functions

Macro Language Interactive Commands

Macro Language Source Code Example

Macro Language Source Code Example

[@k example,, macro language source, source code example,, macro language]

Below is a partial sample macro file source listing taken from the sampling rate conversion macro included with Hypersignal-Macro software. The listing has been simplified for brevity:

```
; parameters passed to this macro from the "Macro Manager" menu are:
; #1: input waveform name
; #2: M; integer amount to multiply input sampling rate by
; #3: N; integer amount to divide input sampling rate by
; #4: output waveform name

; declare and calculate a few variables that will be needed
Integer InSampFreq InLen InFrameSize InCutOffFreq CutOffFreq D
Integer StretchSampFreq StretchLen StretchFrameSize
Integer OutSampFreq OutLen OutFrameSize OutCutOffFreq

InSampFreq = GetHeaderValue(#1, Fs)  ; get values from input header
InLen = GetHeaderValue(#1, Len)
InFramesize = GetHeaderValue(#1, Fr)
InCutOffFreq = InSampFreq div 2      ; "Nyquist" rate for input waveform

StretchSampFreq = InSampFreq * #2    ; calculate "stretched" values
StretchLen = InLen * #2
StretchFramesize = InFramesize * #2

OutSampFreq = StretchSampFreq div #3 ; calculate final converted values
OutLen = StretchLen div #3
```

```
OutFramesize = StretchFramesize div #3
OutCutOffFreq = OutSampFreq div 2    ; "Nyquist" rate for output

Difference Equations                 ; apply difference equation to stretch input
     stretch                         ; output waveform name
     y[n]=x1[n\#2]                   ; #2 is second input parameter
     blank
     blank
     blank
     #1                              ; input waveform
     none
     @StretchLen                     ; length of input waveform * parameter #2
     @StretchSampFreq                ; sampling freq of input * parameter #2
     @StretchFramesize               ; framesize of input * parameter #2
endfunc

if InCutOffFreq<OutCutOffFreq then   ; design FIR filter to interpolate/anti-alias
     CutOffFreq = InCutOffFreq       ; between sample points stretched
else                                 ; waveform--use whichever cutoff
     CutOffFreq = OutCutOffFreq      ; frequency is less
endif

FIR Filter Construction              ; design interpolation anti-aliasing filter
     interp                          ; filter name
     kw                              ; design method (KW = Kaiser Window)
     @StretchSampFreq                ; sampling frequency (in Hz)
     0                               ; center frequency (in Hz)
     @CutOffFreq                     ; bandwidth (in Hz)
     0                               ; left transition bandwidth (in Hz)
     250                             ; right transition bandwidth (in Hz)
     50                              ; stopband attenuation (in dB)
     auto                            ; let software choose number of taps
     1                               ; passband ripple (in dB)
     $RUN                            ; leave menu editor, execute function
     $C                              ; construct filter
endfunc

Convolution/Correlation              ; apply interpolation/anti-aliasing filter
     stretch                         ; input waveform name
     interp                          ; filter name
     stretchf                        ; output waveform name
     –                               ; '–' invokes convolution
endfunc

D = GetHeaderValue(interp, Len)      ; get interp filter length

Difference Equations                 ; use difference equation to decimate
                                     ; output and remove filter delay
     #4                              ; input parameter #4 is output filename
```

```
    y[n]=x1[@D+n*#3]          ; input parameter #3 is divide factor N
    ""                        no entry
    ""
    ""
    stretchf                  ; input waveform "x1"
    none                      ; x2 not used
    @OutLen                   ; final length of output waveform
    @OutSampFreq              ; final sampling freq of output waveform
    @OutFramesize             ; final framesize of output waveform
endfunc
```

Related Macro Language Topics

Macro Language: Hypersignal Programmability

Macro Record Mode

Hypersignal Macro Programs

Creating Macro Language Programs and Functions

Macro Language Variables, Constructs, and Procedures

Macro Language Operators

Macro Language Built-In Functions

Macro Language Interactive Commands

Macro Language Source Code Example

File Import/Export Functions in Hypersignal

[@k importing waveform files, exporting waveform files]

All Hypersignal-Macro series software packages contain a powerful, mature "File Acquisition" function, which has evolved over the years in response to the demands of many different types of customers. This function can acquire binary and ASCII data in decimal, hexadecimal, floating-point, and various DSP manufacturer formats. Data sources include disk files, RS-232 serial channel, and direct PC memory access. Data can be unsigned, μ-Law or A-Law, complex, and multichannel; control or other non-data information contained in individual samples can be stripped. Sample size ranges from 8 bits to 32 bits in 4 bit, an "interleave factor" can be specified when dealing

with multichannel data, the mean can be subtracted from incoming data, and initial data (such as a header from another program) can be discarded from the beginning of the input data. Hypersignal waveform files can also be exported to ASCII decimal, floating-point, and hexadecimal formats.

The File Acquisition function can be invoked from the Utilities menu when Hypersignal is operating in user mode, and from the f_fileac macro language function when Hypersignal is being operated in engine mode.

Import/Conversion Topics

File Import/Export Functions in Hypersignal

Tim2Wav and Wav2Tim Waveform File Conversion Functions in Hypersignal

Tim2Wav and Wav2Tim Waveform File Conversion Functions in Hypersignal

[@k conversion,, .WAV files, conversion,, .TIM files]

The tim2wav.mac and wav2tim.mac programs are included with all Hypersignal-Macro series software packages, and can be invoked from the Macro Manager menu when Hypersignal is operating in user mode. The tim2wav.mac and wav2tim.mac programs call the f_tm2wv and f_wv2tm functions, respectively. Use the 'F3' key to enter the Macro Manager menu, and then enter the program name. Both programs prompt for input and output waveform filenames, with no extension.

The f_tm2wv and f_wv2tm functions can also be called from DSPower-Block Diagram when Hypersignal is operating in engine mode.

The f_tm2wv and f_wv2tm functions convert waveform files between Hypersignal .TIM format and Microsoft .WAV format.

Regardless of operating mode, all Hypersignal time domain functions, including Difference Equations, Convolution, FFT, D/A playback, etc. can read .WAV files, either 8-bit or 16-bit, directly as input. For these

functions, a '.WAV' extension must be entered or supplied explicitly to the appropriate input waveform name fields. *The only exception* is waveform editing with 8-bit .WAV files. Only 16-bit .WAV files are allowed in Hypersignal waveform editing.

When operating Hypersignal in user mode, if you need to reconvert a .WAV file after processing by Hypersignal, or convert any other Hypersignal waveform file to .WAV format, use the Hypersignal tim2wav.mac program, as explained above.

NOTE: At no time do Hypersignal functions generate .WAV files automatically as output, including the PutHeaderValue and PutDataValue macro language commands. This is because the .WAV format file header is limited, and cannot support all the parameters output in typical Hypersignal waveform files. It is possible to add application-specific information to .WAV files, but this capability is not encouraged in the Microsoft documentation, nor is it found in widespread use. For more information about these limitations, and the .WAV file format in general, see Signalogic DSP Software System Guide section 4.2, Microsoft .WAV Waveform File Format, and chapter 8 in the Microsoft Windows Multimedia Programmer's Reference.

File Import/Conversion Topics

File Import/Export Functions in Hypersignal

Tim2Wav and Wav2Tim Waveform File Conversion Functions in Hypersignal

External Program Control In Hypersignal

[@k controlling external programs, external programs]

External programs can be executed from a Hypersignal-Macro series program in three ways:

1. From one of six (6) different user-defined menu labels when Hypersignal is operating in user mode. These menu labels can be customized and associated with various external programs using the Hypersignal the Utilities menu selection Ext Program Config.

2. From macro language programs which have been started from the Macro Manager menu when Hypersignal is operating in user mode. This technique is used by numerous macro programs included with Hypersignal-Macro EX and Hypersignal-Acoustic software, which need to execute protected-mode DOS programs in several cases.

3. From macro language programs which have been started by another application when Hypersignal is operating in engine mode. This technique is used by DSPower-Block Diagram when using a Hypersignal package as a DSP Engine

Real-mode or protected-mode DOS programs can be executed in all cases, and Windows programs can be executed if Hypersignal is being operated in engine mode by a Windows application. In the latter two techniques listed above, up to 10 command-line parameters can be passed from the macro language program to the external program.

Related Hypersignal System Consideration Topics

External Program Control in Hypersignal

User-Interface in Hypersignal

Online Help in Hypersignal

Graphics and Mouse Support in Hypersignal

80x87 Math Coprocessor Support in Hypersignal

Printer Support in Hypersignal

Documentation Supplied with Hypersignal

User-Interface in Hypersignal

[@k Hypersignal user-interface]

The Hypersignal-Macro series user-interface contains many aides and utilities to facilitate use of the program. Moreover, the maturity of the software reflects the countless suggestions and criticisms made by customers over the past few years. All functions and displays are part of the same integrated environment, and produce and accept compatible waveform data. The basic interface is menu-driven, with individual functions and displays being controlled by "set-up

parameters". This approach has enabled Hypersignal to gain popularity and success as a "lab-bench" tool known for its ease-of-use and environment conducive to freedom-of-thought.

In engine mode, all visual aspects of the Hypersignal user-interface are deactivated. In engine mode, all menus, data-entry fields, etc. receive simulated keyboard input from the macro language "external application pipe" facility (see Pipe.386 VxD in section 1.1, Relationship of Signalogic DSP Software Components).

Related Hypersignal System Consideration Topics

External Program Control in Hypersignal

User-Interface in Hypersignal

Online Help in Hypersignal

Graphics and Mouse Support in Hypersignal

80x87 Math Coprocessor Support in Hypersignal

Printer Support in Hypersignal

Documentation Supplied with Hypersignal

Online Help in Hypersignal

[@k help,, online]

When operating a Hypersignal-Macro series software package in user mode, a native online Help facility provides over 300 pages of online Help, selectable from menus and displays in the software. The native Hypersignal online Help facility is context-sensitive, refers to appropriate sections in the documentation, and can be easily controlled using the keyboard or mouse.

Related Hypersignal System Consideration Topics

External Program Control in Hypersignal

User-Interface in Hypersignal

Online Help in Hypersignal

Graphics and Mouse Support in Hypersignal

80x87 Math Coprocessor Support in Hypersignal

Printer Support in Hypersignal

Documentation Supplied with Hypersignal

Graphics and Mouse Support in Hypersignal

When operating a Hypersignal-Macro series software package in user mode, over 100 different video hardware types, including CGA, EGA, VGA, Hercules, various 800x600 and 1024x768 video boards, and several specialized hardware types and resolutions, are supported with the MetaWindow graphics drivers. Support is also available for NEC PC98xx-series machines used in Japan. The MetaWindow drivers also include support for several types of mice, either through the mouse manufacturer's memory-resident driver or directly through the serial port. All Hypersignal menus and displays can be easily controlled using either the mouse or keyboard. A "hardware probe" is conducted by the software to provide best-guesses for graphics and mouse driver selection.

When a Hypersignal-Macro series software packge is being operated in DSP Engine mode, the MetaWindow drivers are not loaded.

Related Hypersignal System Consideration Topics

External Program Control in Hypersignal

User-Interface in Hypersignal

Online Help in Hypersignal

Graphics and Mouse Support in Hypersignal

80x87 Math Coprocessor Support in Hypersignal

Printer Support in Hypersignal

Documentation Supplied with Hypersignal

80x87 Math Coprocessor Support in Hypersignal

Hypersignal-Macro series software packages contain automatic detection and utilization for 80x87 math processors. The automatic detection process can be manually overridden if necessary (see section 3.2.5, Installing and Configuring Hypersignal Software for User Mode Operation. A math processor in your system is strongly recommended: Hypersignal-Macro series software packages use 8-byte and 10-byte precision in several instances, and the presence of a math processor will increase the speed of the software in general and several design and simulation functions, such as filter design and difference equations, in particular. Speed increases can be substantial (from 4 to 6 times). If no math processor is present in your system, Hypersignal-Macro series software packages will exactly emulate all 80x87 calculations using software routines, producing no difference in operating results (but executing much slower).

Related Hypersignal System Consideration Topics

External Program Control in Hypersignal

User-Interface in Hypersignal

Online Help in Hypersignal

Graphics and Mouse Support in Hypersignal

80x87 Math Coprocessor Support in Hypersignal

Printer Support in Hypersignal

Documentation Supplied with Hypersignal

Printer Support in Hypersignal

Hypersignal-Macro series software packages contain built-in support for about 200 dot-addressable printers, ranging from various Epson and compatibles to LaserJet and compatibles. When being operated in user mode, waveform and instrumentation displays in Hypersignal software can be printed as screen dumps or as publication-quality EMS memory regenerations. Controls include print image dimensions,

orientation, background, printer resolution (dpi), and number of copies.

Related Hypersignal System Consideration Topics

External Program Control in Hypersignal

User-Interface in Hypersignal

Online Help in Hypersignal

Graphics and Mouse Support in Hypersignal

80x87 Math Coprocessor Support in Hypersignal

Printer Support in Hypersignal

Documentation Supplied with Hypersignal

Documentation Supplied with Hypersignal

Hypersignal-Macro series software documentation is composed of Users Manual, Hardware Reference Guide, Software Reference Guide, and Tutorial, totaling more than 1200 pages. The Tutorial is illustrated with numerous menu and waveform display screen examples. The Hardware Reference Guide discusses all DSP/acquisition hardware and systems supported by the software individually and in detail, and also provides standardized analog and performance specification tables for each hardware type.

Related Hypersignal System Consideration Topics

External Program Control in Hypersignal

User-Interface in Hypersignal

Online Help in Hypersignal

Graphics and Mouse Support in Hypersignal

80x87 Math Coprocessor Support in Hypersignal

1.2.2 DSPower-HWLib Software Summary

DSPower-HWLib software consists of two (3) DLLs and various C/C++ source code examples, include files, online Help files, and support programs. The DSPower-HWLib DLL itself is a Windows DLL that offers a C/C++ interface to supported hardware. Low-level functions include board and individual processor control, program file download, memory transfer, and semaphore management. High-level functions include acquisition to waveform file, D/A playback from file or continuous function generation, stimulus & response, various DSP functions, and more.

DSPower-HWLib software also includes the Engine Manager and Hardware Manager DLLs (enmgr.dll and hwmgr.dll), which are used to control DSP Engines, and maintain current configuration for supported DSP/analog hardware types, respectively. See Signalogic DSP Software Users Guide chapters 7 and 8, Using Engine Manager and Using Hardware Manager.

The Engine Manager allows multiple copies of DSP Engines to be opened and controlled. Arbitrary engine functions can be executed using the DSExecEngineFunction call.

Hardware Manager maintains a Hardware Setup File which includes such parameters as I/O and memory base addresses, analog I/O voltage levels, memory amounts and architecture, processor clock rates, and more. It also offers the DSShowHardwareSelectorDlg call, which displays a Hardware Selector dialog box allowing user-specified selection of hardware type and configuration parameters.

1.2.3 DSP Source Code Interface Software Summary

The DSP Source Code Interfaces provide access to DSP source for nine (9) different DSP devices, including math and DSP routines, analog I/O drivers, and real-time processing structures used by the

Hypersignal packages. They can be used to customize Hypersignal operation, or operation of some DSPower-HWLib software high-level operations when a Hypersignal package is used as the DSP Engine.

Modification of, or addition to, the DSP Source Code Interface for a specific DSP chip allows customization of the standard executable DSP files delivered with Hypersignal. The functions and algorithms used by Hypersignal--either in user mode, or when used as a DSP engine with the DSPower series software--can be used in intact, or modified to fit application-specific situations, while still maintaining standard Hypersignal-Macro and/or DSPower series sofware functionality.

A project or product can be partially or wholly designed around the basic real-time interrupt structure and DSP and math source code routines available, allowing real-time development directly in the PC environment. By incorporating the real-time disk and video accessibility built-in to the source code interfaces, and using the built-in Hypersignal download function (which accept DSP manufacturer linker output directly), your problem reduces to simply writing and generating the DSP source code specific to your algorithms. You can rely on Hypersignal to allow you to monitor results in real-time, provide you with device drivers to the analog I/O resources of the DSP/acquisition hardware, provide background and foreground interrupt processing, and allow your data to be transferred continuously to the host computer disk and/or instrumentation displays. The time saved in not dealing with the intricacies of the DSP/acquisition hardware board-host computer interface, the complexity of continuous, real-time transfer of data over the host computer bus, and the disk and instrumentation display housekeeping will be measured in months.

Currently the Analog Devices ADSP-21xx, AT&T DSP32C and DSP3210, Motorola DSP5600x and DSP96002, NEC 77220, and Texas Instruments TMS320C25, TMS320C3x, TMS320C5x, and TMS320C4x devices are supported by DSP Source Code Interfaces.

DSP Source Code Interfaces: More Detail

The DSP Source Code Interfaces include math and DSP library routines, device drivers for analog resources on supported boards, and

real-time, interrupt-driven structures that allow foreground and background processing in conjunction with analog I/O sample interrupts. Without modification, each DSP Source Code Interface forms the basis for standard Hypersignal operation, including instrumentation displays, continuous transfer to/from disk, and file-based acceleration. In other words, if you were to assemble and link the standard DSP source code for each supported DSP chip, without any changes, you would end up with the executable files that are sent with every Hypersignal real-time software package. However, should you then modify, add, or otherwise customize the source code, you could create new executable files that would customize your system's operation.

Changes can be simple, such as interfacing to new analog converters or accessing other resources on the board, or extensive, such as simulating and testing complete algorithms or products. Because the mechanism for transferring data continuously and in real-time between high level Hypersignal functions and the DSP is transparent, you are free to concentrate on the DSP problem at hand. By simply placing the data you need to see or monitor in the correct buffer, at such time or intervals as it becomes ready, you can completely control the content and operation of high level Hypersignal instrumentation and disk transfer functions. Although not necessary, maintaining the Hypersignal infrastructure for transferring data and communicating control parameters offers many advantages throughout the course of your product development. Examples include comparing input data and algorithm output data in real-time, leaving "field-test" capability embedded in your product, and evaluating real-time limits.

For each DSP chip, the built-in Hypersignal download functions accept direct linker output from the manufacturer's development tools. The process of making source code changes and sending those changes into real-time action is a short one:

1) make source code changes using editor
2) compile and/or assemble
3) link (creating new executable file)
4) run Hypersignal

This process is fully documented for each supported DSP device family in Signalogic DSP Software Users Guide chapter 5, Using DSP Source Code Interface Software. Also, DSP source code files are well-commented and clearly point out variables, structures, and procedures related to the intrinsic Hypersignal high level interface. Additionally,

the Hypersignal-Macro Series Hardware Reference Guide includes detailed and extensive information on individual DSP/acquisition hardware types supported.

Related Topics

DSP Source Code Interfaces: More Detail

DSP Source Code Interface Partial Function List

DSP Source Code Interface Partial Function List

Below is a partial list of typical source code and utility files included with each DSP Source Code Interface:

xxxDef	variable and constant definitions (xxx refers to DSP chip family)
xxxBEG	DSP chip initialization code, analog I/O and converter initialization code, ISR vector initialization, variable initialization, and initialization of high level interface (xxx refers to DSP chip)
xxxLINK	linker specification and map control file (xxx refers to DSP chip)
ASM	batch file to assemble typical assembly language source code files
LNK	batch file to link typical configuration
ISRn	various ISRs (interrupt service routines) for different analog I/O functions; each ISR contains different entry points for specific DSP/acquisition hardware that isolate hardware-dependent analog I/O code and which converge into common, highly optimized code
ARCTAN	2-quadrant arctangent approximation
CONVLV	convolution/correlation
DIVIDE	fractional and integer divide for fixed-point DSPs; floating-point divide for floating-point DSPs
EXP2	base 2 exponentiation

FFT	complex input, decimation-in-time FFT/inverse FFT
FIR	FIR filter implementation
IIR2	IIR filter implementation (modified direct form using 2nd-order, or "biquad cascade" sections)
IIR3	IIR filter implementation (balanced pole-zero form using 2nd-order, or "biquad cascade" sections)
LOG2	base 2 log approximation
MAG	complex input magnitude
MAX	location of maximum value in buffer
MIN	location of minimum value in buffer
PHASE	complex input 4-quadrant phase approximation
SQRT	squareroot approximation
WINDOW	time domain window application and zero-filling
XFER	complex-ratio transfer function calculation

Related Topics

DSP Source Code Interfaces: More Detail

DSP Source Code Interface Partial Function List

1.2.4 DSPower-Block Diagram Software Summary

DSPower® Block-Diagram is a Windows program with block-diagram and graphical control user interfaces; depending on the "DSP Engine(s)" with which it is used, DSPower offers math/DSP simulation and real-time execution of block-diagrams, interactive instrumentation, measurement, displays, visualization, and analysis, data acquisition, and source code generation.

1.2.5 DSPower-C/C++ Code Generator Software Summary

This section not available yet.

1.2.6 Real-Time Convolver Software Summary

The Real-Time Convolver™ is an add-on program intended for use with Hypersignal-Acoustic. It offers FFT-based real-time convolution of analog input data, or accelerated processing of file data, using impulse responses from various sources.

The Real-Time Convolver allows long impulse responses (filters) to be applied to sound inputs in real-time. Depending on the DSP/analog hardware used, real-time sampling rates range from 8 kHz to 48 kHz. A mode is also available that processes sound files. The Real-Time Convolver allows either mono or stereo operation with one or two different filters. Filters can be supplied from outside sources or created with Hypersignal-Acoustic's built-in Difference Equations and FIR/IIR Filter Design functions.

Typical applications include 3-D sound, various real-time sound simulators, precise real-time equalization, and impulse responses gathered from on-site or scale-model acoustic measurements or created by architectural "imaging" programs.

Currently, the Real-Time Convolver can be obtained from Signalogic packaged with a wide range of off-the-shelf DSP/analog hardware, including Ariel, CAC, National Instruments, and Signalogic boards. When combined with Signalogic's convolution algorithms, these boards offer performance comparable to dedicated convolution instruments, but at a fraction of the cost. For example, with the National Instruments AT-DSP2200, which contains the 32-bit floating-point AT&T DSP32C device, two channels of analog I/O with 92 dB typical dynamic range, and 384k x 32 of zero wait-state SRAM, a 10,000 pt. filter can be applied in real-time to stereo sound data at 44.1 kHz. At lower rates, or with mono operation, filter lengths up to 25,000 pts. are possible. In all cases, a higher amount of DSP chip

external memory (SRAM or DRAM) on the board translates to longer filter lengths.

The Real-Time Convolver package requires Hypersignal-Acoustic software, a supported DSP/analog hardware type, DOS 5.0 or higher, and at least a 33 MHz 80386 machine with 4 Mb or more of memory.

1.3 System Requirements

System requirements for Signalogic software components are listed below. If differences exist between software components, requirements may be listed by product name. For more detail on Hypersignal-Macro series software, see Signalogic DSP Software System Guide section 1.2.1, Hypersignal-Macro Series Software Summary; for more detail on DSPower-HWLib software, see Signalogic DSP Software System Guide section 1.2.2, DSPower-HWLib Software Summary; for more detail on DSPower-Block Diagram software, see Signalogic DSP Software System Guide section 1.2.4, DSPower-Block Diagram Software Summary.

System Requirements Topics

Operating System Requirements

Hypersignal-Macro, Hypersignal-Macro NRT

DOS version 3.3 or higher

Hypersignal-Macro EX, Hypersignal-Acoustic

DOS version 5.0 or higher

All DSPower series packages

DOS version 5.0 or higher
Windows version 3.1 or higher

Additional System Requirements Topics

Operating System Requirements

Processor Requirements
Memory (RAM) Requirements
Video / Graphics Requirements
Mouse Requirements
Math Coprocessor Requirements
Printer Requirements
DSP Engine Requirements
DSP/Analog Hardware Requirements
Windows C/C++ Development Tool Requirements
DSP Development Tool Requirements

Processor Requirements

Hypersignal-Macro series

33 MHz 80386 or faster

DSPower-Block Diagram

33 MHz 80486 (66 MHz or faster is strongly recommended)

Additional System Requirements Topics

Operating System Requirements
Processor Requirements
Memory (RAM) Requirements
Video / Graphics Requirements
Mouse Requirements
Math Coprocessor Requirements
Printer Requirements
DSP Engine Requirements
DSP/Analog Hardware Requirements
Windows C/C++ Development Tool Requirements
DSP Development Tool Requirements

Memory (RAM) Requirements

Hypersignal-Macro, Hypersignal-Macro NRT

448 kbyte free conventional memory needed, 1 Mbyte expanded memory (EMM) needed for publication-quality print

Hypersignal-Macro EX, Hypersignal-Acoustic

same as Hypersignal-Macro, but 4 Mbyte system RAM needed for long FFT, long convolution, long correlation, other macro functions

DSPower-Block Diagram

4 Mbyte, 8 Mbyte strongly recommended

Additional System Requirements Topics

Operating System Requirements

Processor Requirements

Memory (RAM) Requirements

Video / Graphics Requirements

Mouse Requirements

Math Coprocessor Requirements

Printer Requirements

DSP Engine Requirements

DSP/Analog Hardware Requirements

Windows C/C++ Development Tool Requirements

DSP Development Tool Requirements

Video / Graphics Requirements

Hypersignal-Macro series

Any of over 100 different types of hardware supported, including CGA, EGA, VGA, Hercules, AT&T, and popular 800x600 and 1024x768 formats. If you are unsure of selection to make in ***Graphics/Mouse Config*** menu, try following: 42 (VGA), 65 (popular 800x600), 76 (popular 800x600), 95 (8514 VESA compatible).

DSPower series

Any video/display hardware currently installed under Program Manager ***Main*** group, ***Windows Setup*** item (see "Change System Settings" for list). A 24-bit or 32-bit video board capable of fast, hardware-based bitmap updating is strongly recommended. See Signalogic DSP Software System Guide section 1.4.2, Video and Display Considerations, for more information.

Additional System Requirements Topics

Operating System Requirements

Processor Requirements

Memory (RAM) Requirements

Video / Graphics Requirements

Mouse Requirements

Math Coprocessor Requirements

Printer Requirements

DSP Engine Requirements

DSP/Analog Hardware Requirements

Windows C/C++ Development Tool Requirements

DSP Development Tool Requirements

Mouse Requirements

Hypersignal-Macro series

Interface to memory-resident mouse driver supported; some mice supported directly through serial ports.

DSPower series

Any currently installed inside the Program Manager | Main | Control Panel | Mouse dialog box.

Additional System Requirements Topics

Operating System Requirements

Processor Requirements
Memory (RAM) Requirements
Video / Graphics Requirements
Mouse Requirements
Math Coprocessor Requirements
Printer Requirements
DSP Engine Requirements
DSP/Analog Hardware Requirements
Windows C/C++ Development Tool Requirements
DSP Development Tool Requirements

Math Coprocessor Requirements

All packages

80387 math coprocessor automatically detected and utilized if installed in system. 80486 and higher always used.

Additional System Requirements Topics

Operating System Requirements
Processor Requirements
Memory (RAM) Requirements
Video / Graphics Requirements
Mouse Requirements
Math Coprocessor Requirements
Printer Requirements
DSP Engine Requirements
DSP/Analog Hardware Requirements
Windows C/C++ Development Tool Requirements
DSP Development Tool Requirements

Printer Requirements

Hypersignal-Macro series

Screen-dump and publication-quality print to over 185 printers supported. An appropriate amount of expanded memory (minimum 1 Mbyte) is required in the system to allow publication-quality printing of waveform displays.

DSPower series

Any currently installed inside the Program Manager | Main | Control Panel | Printer dialog box.

Additional System Requirements Topics

Operating System Requirements

Processor Requirements

Memory (RAM) Requirements

Video / Graphics Requirements

Mouse Requirements

Math Coprocessor Requirements

Printer Requirements

DSP Engine Requirements

DSP/Analog Hardware Requirements

Windows C/C++ Development Tool Requirements

DSP Development Tool Requirements

DSP Engine Requirements

For information about DSP Engines, see Signalogic DSP Software System Guide section 1.1, Relationship of Signalogic DSP Software Components.

DSPower-Block Diagram

Hypersignal-Macro, -Macro EX, or -Acoustic v4.0 or higher; MATLAB v4.0 or higher

DSPower-HWLib DLL

Hypersignal-Macro, -Macro EX, or -Acoustic v4.0 or higher;
mmsystem.dll (for Windows 3.1 or higher)

Additional System Requirements Topics

Operating System Requirements

Processor Requirements

Memory (RAM) Requirements

Video / Graphics Requirements

Mouse Requirements

Math Coprocessor Requirements

Printer Requirements

DSP Engine Requirements

DSP/Analog Hardware Requirements

Windows C/C++ Development Tool Requirements

DSP Development Tool Requirements

DSP/Analog Hardware Requirements

A supported DSP/analog or DSP/acquisition hardware type (e.g. a plug-in ISA board) is required in the following two (2) cases:

1) When a Hypersignal-Macro, Hypersignal-Macro EX, or Hypersignal-Acoustic package is to be used.

2) When a DSPower-Block Diagram or DSPower-HWLib package is to be used in conjunction with a Hypersignal package as the DSP Engine.

See section 1.4.3, DSP/Analog Hardware Considerations, for more information.

Additional System Requirements Topics

Operating System Requirements

Processor Requirements

Memory (RAM) Requirements
Video / Graphics Requirements
Mouse Requirements
Math Coprocessor Requirements
Printer Requirements
DSP Engine Requirements
DSP/Analog Hardware Requirements
Windows C/C++ Development Tool Requirements
DSP Development Tool Requirements

Windows C/C++ Development Tool Requirements

A Windows C/C++ compiler is currently necessary in two (2) cases:

1) If you have the DSPower-HWLib or DSPower-SPLib packages, which contains C/C++ source code examples. These examples illustrate use of the DSPower-HWLib and DSPower-SPLib DLLs by making calls to DLL functions.

2) If you intend to use the C source code blocks included with DSPower-Block Diagram. Examples include the C source code version of the Difference Equations block.

In these cases, the following Windows C/C++ compilers are suggested:

Borland C/C++ v3.1 or higher
Microsoft Visual C/C++ v1.5 or higher
WATCOM C/C++^{32} v9.5a or higher

Additional System Requirements Topics

Operating System Requirements
Processor Requirements
Memory (RAM) Requirements
Video / Graphics Requirements
Mouse Requirements

Math Coprocessor Requirements
Printer Requirements
DSP Engine Requirements
DSP/Analog Hardware Requirements
Windows C/C++ Development Tool Requirements
DSP Development Tool Requirements

DSP Development Tool Requirements

TMS320C25 Source Code Interface

Texas Instruments "DSP COFF Assembler", v5.05 or higher, Texas Instruments "DSP COFF Linker", v5.05 or higher

DSP5600x Source Code Interface

Motorola "DSP56000 Assembler", v4.1 or higher, Motorola "DSP Linker", v4.1 or higher

DSP32C Source Code Interface

AT&T v1.3.1 "d3as Assembler", v1.3.1 or higher, "d3ld Linker", v1.3.1 or higher

TMS320C3x Source Code Interface

Texas Instruments "TMS320C3x/C4x COFF Assembler", v4.5 or higher, Texas Instruments "TMS320C3x/C4x COFF Linker", v4.5 or higher

ADSP-21xx Source Code Interface

Analog Devices "ASM21" assembler, v1.13 or higher, Analog Devices "LNK21" linker, v1.12 or higher

DSP32C Source Code Interface

AT&T v1.3.1 "d3as.exe" assembler, v1.3.1 "d3ld.exe" linker

DSP32C Source Code Interface

AT&T v1.3.1 "d3as.exe" assembler, v1.3.1 "d3ld.exe" linker

DSP32C Source Code Interface

AT&T v1.3.1 "d3as.exe" assembler, v1.3.1 "d3ld.exe" linker

DSP32C Source Code Interface

AT&T v1.3.1 "d3as.exe" assembler, v1.3.1 "d3ld.exe" linker

Additional System Requirements Topics

Operating System Requirements

Processor Requirements

Memory (RAM) Requirements

Video / Graphics Requirements

Mouse Requirements

Math Coprocessor Requirements

Printer Requirements

DSP Engine Requirements

DSP/Analog Hardware Requirements

Windows C/C++ Development Tool Requirements

DSP Development Tool Requirements

1.4 System Considerations

Signalogic DSP Software packages have been designed to run on IBM 386, 486, and Pentium machines using Microsoft Windows v3.1 or higher. Hypersignal-Macro series software packages have also been designed to run stand-alone under DOS v3.3 or higher with over 100 different types of graphics hardware and several different types of mice installed.

Signalogic DSP Software packages will likely function correctly on many "IBM-compatible" or similar machines with compatible or similar graphics hardware. However, the word "compatible" means something different to every hardware and software manufacturer, and Signalogic can make no guarantee concerning compatibility.

One area that has a widespread reputation for being a source of problems is video drivers. See Signalogic DSP Software System Guide section 1.4.3, Video Considerations, for more information.

1.4.1 Memory Considerations

[@j conventional memory, virtual memory, DOS extender]

This section covers memory considerations of all types, including conventional memory, virtual memory, differences when used in protected-mode vs. real-mode, and typical DSP-oriented function use of available memory. Both Hypersignal-Macro series software and DSPower series software issues are covered.

Hypersignal Memory Considerations

The standard Hypersignal core functions exclusively utilize "conventional DOS" memory, and use overlays to virtualize this area for both program and data information. Some of the new processing functions added to Hypersignal-Macro EX and Hypersignal-Acoustic have been implemented as protected-mode programs controlled by macro functions. Many of these functions require large amounts of program and data memory. Examples include long FFT, convolution, and correlation, MLS stimulus & response, etc. You should have at least 8 Mbyte of extended memory in your system if you want to use these functions "comfortably", without causing virtual memory disk swaps.

The DOS extender program DOS4GW.EXE is used to load and run the protected-mode programs. Once running, the protected-mode programs request extended memory based on processing needs. Some important notes about amount of extended memory and its usage:

1) The amount of memory installed in your machine is not the amount available. The first 1 Mbyte is devoted to "conventional memory" and housekeeping machine requirements. Also, other programs use extended memory, such as SMARTDRV.EXE, which is commonly installed with later versions of DOS and Windows, and which takes up 2 Mbyte of extended memory in its default mode. Finally, expanded memory may take a chunk, too--although expanded memory is used less often with state-of-the-art systems.

To determine if your system is configured to use some portion of extended memory as the older "expanded" type of memory, look for the line "device=EMM386.EXE ..." in your system's CONFIG.SYS file. If this line exists, but does not contain the attribute "NOEMS",

then it some amount of expanded memory may be configured. See DOS or Windows manuals for more information.

Thus, a machine with only 4 Mbyte of extended memory often ends up with not much available for user programs. You can run the PMINFO.EXE program included with Hypersignal to determine the amount of extended memory actually available.

2) The protected-mode programs can be run under Windows (i.e. if Hypersignal has been launched directly from Windows, or run from a DOS command prompt that was launched from Windows). When run under Windows, the protected-mode program's use of system memory is fully compatible with Windows and whatever Windows applications are already running, including access to Windows virtual memory, which extends the amount of memory available on a system by swapping unused sections to disk file. ***However, do not expect virtual memory to help with typical DSP programs.*** For example, lcorr.exe, lconv.exe, and lfft.exe, are exceedingly slow when they use virtual memory, because the bit-reversed memory-access nature of the FFT routine used by these programs causes excessive "disk thrashing". Disk thrashing occurs when virtual memory is being swapped in and out from disk file so frequently that programs that should take seconds end up taking hours. To run larger FFT sizes, larger filter sizes, and larger MLS stimulus & response sizes, it is strongly suggested that you upgrade (expand) the amount of memory in your computer. For example, on a typical 486 system running Windows 3.1, and with SMARTDRV.EXE installed in the AUTOEXEC.BAT file and no EMS (expanded) memory installed, 8 Mbyte of extended memory typically would allow a 262,144 pt. complex FFT in the lfft.mac function without disk thrashing. But, using the lcorr.mac or lconv.mac functions with an implied FFT size of 262,144 (i.e. a filter size of 200,000 pts.) would require a full 16 Mbyte, due to the extra memory needed for convolution and correlation.

A note about DOS versions less than 5.0: DOS4GW.EXE still needs some conventional memory to get started--before loading a protected-mode program--and this can be a problem with DOS versions less than 5.0. In these cases, the combination of Hypersignal, the MET_SHEL.EXE graphics driver, DOS's command.com, and other utilities can occupy enough conventional memory to keep DOS4GW.EXE from running. When this happens, you will see "not enough memory" error messages when attempting to run the extended macro functions. Here are some suggestions to deal with this problem:

- use DOS 5.0 or higher, and make sure the DOS=HIGH,UMB command is in your CONFIG.SYS file
- run PMINFO.EXE and RMINFO.EXE, included with DOS4GW.EXE, to diagnose the current memory state and protected-mode compatibility of your machine, and make sure there's not a basic problem with switching into protected mode
- start "trimming" your system: REM-out CONFIG.SYS and AUTOEXEC.BAT command lines such as FASTOPEN and SMARTDRV, and remove other memory-resident utilities that take up conventional memory

Note that even for DOS 3.3x through 4.0x this is not a problem when running Hypersignal software in engine mode, because the MET_SHEL.EXE graphics driver is not loaded in this case, thus freeing up an additional 120k of conventional memory. Note: DSPower-Block Diagram software invokes Hypersignal-Macro series software packages in engine mode when executing Hypersignal macro language blocks.

Memory Requirements Topics

Hypersignal Memory Considerations

DSPower Memory Considerations

DSPower Memory Considerations

As with most Windows applications, memory is the critical issue for DSPower-series software performance. Below are some system memory recommendations and comments regarding use of DSPower-series software:

- install at least 8 Mbyte of system memory (see also section 1.4.1, Memory Considerations)
- avoid active use of known memory hog applications at the same time as DSPower-Block Diagram; for example, Microsoft Word
- *do not* set up your system for virtual memory beyond the recommended maximum size, and do not enable 32-bit disk

access (see "386 Enhanced" icon in Program Manager | Main | Control Panel). Both of these settings can decrease rather than increase performance. Especially in the case of DSP or math functions using very large operation sizes (e.g. 262,144 pt. FFT size), extra virtual memory can shut a system down, because the math or DSP function will attempt to use this memory as it would system memory. Examples include the lfft.mac Hypersignal protected-mode DOS program, and DSPower-Block Diagram wavelet transform function.

Memory Requirements Topics

Hypersignal Memory Considerations

DSPower Memory Considerations

1.4.2 Performance Considerations

This section provides some suggestions for achieving and maintaining maximum system performance when using Hypersignal-Macro series and/or DSPower series software. For additional information , see also chapter 5, Troubleshooting.

Hypersignal Performance Considerations

[@j DOS box, protected-mode, real-mode, .pif files]

The highest possible performance for Hypersignal packages is made possible when they are run from a DOS prompt, *and Windows is not installed*. In this mode, called real-mode, Hypersignal installs its own mouse, video and printer drivers; the only possible problems are hardware incompatibility or not enough available conventional DOS memory. If a Hypersignal package fails to load from a DOS prompt, try removing less-than-absolutely-necessary utilities and memory resident programs from your CONFIG.SYS and AUTOEXEC.BAT files, one by one.

If Hypersignal is executed from Windows (either from a DOS prompt started from Windows, or using the Hypersignal icons included with the software), then it will be running in a "DOS box", which simulates a stand-alone DOS environment. This is only a simulation; the host machine is actually not in real-mode, but in protected mode, which is only possible with 386 and 486 processors. When a DOS program is executed under Windows, "time-slices" are always allocated to Windows, even though Windows is in the background (not visible). Thus, the potential exists for Hypersignal, and all its functions, displays, and hardware-based instrumentation, to run slower than if truly stand-alone (in real-mode). To minimize slowdown when running Hypersignal under Windows, take the following steps:

- install Hypersignal-Macro or Hypersignal-Acoustic with an icon in Program Manager (see Runing Hypersignal in User Mode Under Windows, in section 3.2.5, Installing and Configuring Hypersignal for User Mode Operation). This will cause automatic use of hs.pif, hsm.pif, or hsa.pif files included on the Hypersignal distribution diskettes which are optimized for high performance under Windows.
- decrease time-slices given to Windows when in background mode to the minimum value of 1. To do this, select Program Manager | Main | Control Panel | 386 Enhanced.

Even with the above measures in place, you can still expect an approximate 10% slowdown in Hypersignal performance when running under Windows (in protected-mode) as opposed to stand-alone (real-mode). This is the nature of protected mode, and is true for all DOS programs.

NOTE: When installing Hypersignal with the Signalogic Software Setup Utility, which is included with DSPower, the above steps and performance increasing measures, including installation of Hypersignal icons under Windows, are performed automatically.

The Hypersignal .pif files mentioned above are set to optimize performance of Hypersignal. However, if you wish to create your own .pif files, the following settings are recommended:

- operation set to full-screen
- emulate text mode
- start in text mode (video memory)

- desire at least 384 kbyte of memory (maximum used is 640k)
- use high memory
- close window on exit
- no XMS, EMS, or locked memory is needed
- priority and background vs. exclusive operation are not important
- detect Idle Mode should be turned off (VERY IMPORTANT FOR PERFORMANCE)

NOTE: When using Hypersignal with DSPower-Block Diagram, .DSPower uses .pif settings from the files "hsmacro.pif" and "hsacous.pif" located on the DSPower subdirectory. These files should not be edited or changed under any circumstances.

Performance Considerations Topics

Hypersignal Performance Considerations

DSPower Performance Considerations

DSP/Analog Hardware Performance Considerations

DSPower Performance Considerations

DSPower-series software performance issues can be divided into three categories:

1. System memory.
2. System video display.
3. System disk access.

Memory is the most critical issue: too small an amount installed, or situations where other active applications have left too little memory available, can bring any Windows application to its knees, including DSPower-series software. Video drivers and display boards, if not optimized and accelerated, can be a source of poor performance, especially in the case of a graphically-oriented application such as DSPower-Block Diagram. The same holds true for disk access, which

is important to DSPower-Block Diagram because of its intensive and constant manipulation of source code files. Currently, the following recommendations are given:

- install at least 8 Mbyte of system memory (see also section 1.4.1, Memory Considerations)
- avoid active use of known memory hog applications at the same time as DSPower-Block Diagram; for example, Microsoft Word
- *do not* set up your system for virtual memory beyond the recommended maximum size, and *do not* enable 32-bit disk access (see "386 Enhanced" icon in Program Manager | Main | Control Panel). Both of these settings can decrease rather than increase performance. Especially in the case of DSP or math functions using very large operation sizes (e.g. 262,144 pt. FFT size), extra virtual memory can shut a system down, because the math or DSP function will attempt to use this memory as it would system memory. Examples include the lfft.mac Hypersignal protected-mode DOS program, and DSPower-Block Diagram wavelet transform function.
- install a video accelerator board with 24-bit or 32-bit access over a local bus, with as much hardware-based support for typical video operations (bitmap painting, line operations, etc.) as possible
- install a disk drive controller with local bus access

Performance Considerations Topics

Hypersignal Performance Considerations

DSPower Performance Considerations

DSP/Analog Hardware Performance Considerations

DSP/Analog Hardware Performance Considerations

One of the most common areas of concern when using DSP/analog hardware is maximum transfer rates to and from hard disk. To

achieve maximum performance, here are some pointers and suggestions:

1. A fast system. Nothing helps like raw talent. A fast processor (for example, a 66 Mhz 486 or better), a fast hard disk, preferably with its controller card located on a VESA local bus or PCI bus, video controller card located on a VESA local bus or PCI bus, and an internal bus speed of at least 8.33 MHz, and possibly higher, will make a big difference. Locating the hard disk controller and video controller cards on a local bus is important, because the DSP/analog hardware should then have exclusive use of the standard ISA bus during acquisition (99% of off-the-shelf DSP/analog hardware are ISA bus plug-in cards).

2. Some network adaptor cards and other peripheral cards can actually slow down a host PC's internal bus (ISA bus) to rates as low as 4.77 MHz because of the way they respond to bus acknowledgement signals. If you suspect your DSP/analog hardware disk performance is too slow, you might try temporarily taking out all other peripheral boards in your system, and re-measuring.

3. Make sure your available hard disk space is not fragmented. Non-contiguous file space can cause delays in continuous acquisition to disk, which could translate into gaps in the data. You can use an application like Norton Desktop to periodically run a disk-defragmenter utility, say once a week. This should ensure large chunks of free, contiguous disk space at any given time. Also, if the software being used allows it, you might try increasing the number of buffers being used; more buffers act as a greater "shock absorber" for delays caused by non-contiguous file space or bad-tracks.

4. ***Do not use disk space compression***; e.g. DoubleSpace, Stacker, etc. These utilities take extra time to compress data before it is actually stored to disk. Normally, this time is not noticed because a big memory cache is enabled for the system's disk I/O, and data is written first to the cache, allowing programs to think the transfer is complete. But, when long data acquisition operations take place, the cache quickly gets full, and data must then be stored to disk at the same rate as it is coming in. At this point, the performance degradation of a compression utility will be quite noticeable.

5. For some systems, continuous disk transfer performance is actually increased if 32-bit disk access is disabled. This option is

located under Program Manager | Main | 386 Enhanced | Virtual Memory.

6. For some systems, continuous disk transfer performance is actually increased if the SMARTDRV.EXE utility is not loaded. This option is usually specified inside the system's AUTOEXEC.BAT file, and sometimes in its CONFIG.SYS file (in the form "device = C:\WINDOWS\SMARTDRV.EXE" or similar). The reason? Because long, continuous disk acquisition operations at high sampling rates eventually fill up the internal memory cache used by SMARTDRV.EXE; afterwards, the advantages of caching are lost as data must leave the cache at same rate it enters. At this point, the overhead of cache processing does not contribute to disk performance, and may detract. Note, however, that disabling SMARTDRV.EXE will have a detrimental effect on the performance of your system overall, especially for word processing, database, communications, and other applications that make extensive random file accesses. If disabling SMARTDRV.EXE turns out to be necessary for the DSP/analog hardware in your system to achieve continuous acquisition at a certain required maximum sampling rate, you should wrap the disabling into a macro program or DSP engine function, so that it can be disabled and re-enabled automatically.

7. You can use the "sliding window" FFT technique to check for discontinous data in a recorded waveform file that you suspect as possibly having gaps. To do this, perform an FFT function (in Hypersignal-Macro series software, use FFT Generation, in DSPower-Block Diagram use the FFTEx function) with the following parameters:

   Framesize of 250
   FFT size of 512 (FFT order = 9)
   Overlap of 50%
   Hamming Window

This will cause the input waveform to be analyzed with a sliding window that will highlight sharp transitions such as likely to be associated with a discontinuity. Showing the results in the 2-D spectrograph (contour) display function; can do two things: 1) display the entired waveform at once, and 2) allow a "look down" into the frequency data from 0 to -70 dB or -80 dB. If there are gaps in the data they should show up as one or more brightly colored vertical lines, and possibly evenly and/or repetitively spaced. The vertical line

occurs because the discontinuity is seen by the FFT analysis as wideband noise; the even and/or repetitive spacing--if present--may represent the internal buffer size used by the acquisition operation.

Performance Considerations Topics

Hypersignal Performance Considerations

DSPower Performance Considerations

DSP/Analog Hardware Performance Considerations

1.4.3 Video and Display Considerations

This section provides some suggestions for achieving and maintaining maximum video and display performance when using Hypersignal-Macro series and/or DSPower series software. For additional information , see also section 4.2, Troubleshooting Video and Display Problems.

Hypersignal Video and Display Considerations

For detailed information, see Hypersignal-Macro Series Users Manual section 2.4, Selecting Graphics and Mouse Drivers.

When Hypersignal packages are run from a DOS prompt, *and Windows is not installed*, Hypersignal installs its own video drivers. This is the simplest situation possible; video driver selections can be made directly from a Hypersignal Utilities menu selection. When running Hypersignal from Windows (either from Hypersignal icons or from a DOS prompt started from Windows), the situation is more complicated. This is mainly because Windows must temporarily deactivate and save the context of its installed video driver, and allow the DOS program to *perceive* it is running stand-alone. Problems then sometimes appear in transition, when video control is transferred from one driver to another. Display drivers commonly used in the Windows environment sometimes exhibit such transition problems with DOS programs. Such problems include losing "sync" or fuzzy display when

going to the DOS mode, or suffering dramatic (and sometimes garish) color changes when returning to the Windows mode. Another observed problem is background "clearing" problems inside Hypersignal, once it is running. See Signalogic DSP Software System Guide section 4.2, Troubleshooting Video and Display Problems, for suggested workarounds to these problems.

Basically, we have found the standard Windows 3.x VGA driver (640x480 resolution) to be reliable, and allow use of Hypersignal either from a DOS shell command line prompt, or from the Hypersignal icons without problems. However, use of the standard Windows 3.x "SuperVGA" driver, or 800x600 and 1024x768 drivers provided by the graphics board manufacturer has not proven to be so reliable. In general, we have found that the manufacturer's drivers are almost always better than the general Windows driver equivalents. Note that any incompatibility or conflict arising from alternating use of Hypersignal video drivers and installed Windows video drivers is avoided when Hypersignal is used with DSPower; in which case the Hypersignal video drivers are not loaded or used.

Video and Display Considerations Topics

Hypersignal Video and Display Considerations

DSPower-Block Diagram Video and Display Considerations

DSPower-Block Diagram Video and Display Considerations

A 24-bit or 32-bit video board is highly recommended. If you are using a video board that has only the standard AT-style ISA connectors, then you do not have more than a 16-bit video board. Typical 24-bit or 32-bit video boards plug into VESA or PCI local bus connectors, in addition to occupying a standard ISA slot. In almost any Windows application, including DSPower-Block Diagram, user-interface functions are video intensive. In DSPower-Block Diagram, video operations are further intensified when one or more waveform displays or instruments are onscreen at the same time.

NOTE: In DSPower-Block Diagram, if you see the cursor "flashing" when it is positioned over an updating waveform display, then you do not have a video board capable of fast, hardware-based

bitmap painting. It is strongly recommended that you obtain a 24-bit or 32-bit video board capable of hardware-based bitmap updates.

Symbols, dialog boxes, and other Windows graphical items used in DSPower-Block Diagram will appear best at video (screen) resolution of 800x600 or higher (1024x768).At 640x480 resolution (i.e. standard VGA), block symbols can appear "sketchy". You can avoid this by disabling the "Object Size Independent of Display Resolution" option in the Worksheet selection of the Options menu. However, if you do this, then blocks will become bigger at 640x480 resolution, and smaller at 1024x768 resolution.

Likewise, dialog boxes, menus, and toolbar icons can appear disproportionately large at 640x480 resolution, but there is no option controlling the appearance of these items. This discussion does not apply to waveform displays and instruments, which scale in a truly automatic fashion independent of video resolution (although text labels in these displays and instruments can become unreadable at small window sizes).

From experience with various 386 and 486 clone machines, we highly suggest that you use the SuperVGA (800x600) or 1024x768 driver provided by the manufacturer of your video board. It has been our experience that the generic SuperVGA driver included with Windows 3.x does not always work reliably with some video boards. Thus, you might want to take the time to install SuperVGA drivers using the setup utility provided on the video board manufacturer's disk, if you have not already done so.

Another suggestion is to avoid more than 256 colors, depending on the video driver. Experience has shown that video driver bugs appear more often and in more severe form when the video memory needed for any given operation increases, which is the case with increasing number of colors.

Video and Display Considerations Topics

Hypersignal Video and Display Considerations

DSPower-Block Diagram Video and Display Considerations

1.4.4 System Drivers and Resident Program Considerations

This section deals with those miscellaneous collection of Windows and DOS system utilities, drivers, and resident programs that lurk in the background--most of the time unnoticed, but sometimes the source of unexplained problems, performance decreases, or bizarre system behavior.

SHARE.EXE

SHARE.EXE is a Microsoft resident program that originated with later versions of MS-DOS, but also has a Windows 3.1x VxD equivalent. SHARE.EXE can be used to govern access to files on the basis of the mode in which they were opened, and who "owns" them. These features were not originally part of the MS-DOS system, and so SHARE.EXE is one of those typical "growths" that have quietly sprung up around MS-DOS and end up in widespread, but not always flawless, use. For example, some Windows applications require SHARE.EXE in order to run correctly (e.g. Microsoft Word), yet other applications cannot run with it installed (e.g. Ventura Publisher). Typically, SHARE.EXE is found somewhere in the system's AUTOEXEC.BAT file; there may also be a reference in the "[386Enh]" section of the Windows system.ini file.

Hypersignal-Macro series and DSPower-series software can run with SHARE.EXE either installed or not installed in your system. Both types of software have been tested extensively with and without SHARE.EXE. If installed, DSPower-Block Diagram takes advantage of SHARE.EXE to prevent simultaneous editing on the same waveform file by two different display/editing functions. If for some reasone, you must remove SHARE.EXE from your system at some period during which you are running DSPower-Block Diagram, avoid editing two different copies of the same waveform file at the same time.

System Drivers and Resident Program Considerations Topics

SHARE.EXE

EMM386.EXE

The original use of EMM386.EXE was to provide system control over something called "expanded memory", which provided a system with access to large amounts of memory through a small (64 kbyte) window located in conventional "high", or upper, system memory (between 640k and 1024k). However, as amounts of standard system memory (also referred to as extended memory) increased, expanded memory usage has almost disappeared, because it is too slow, requires the extra cost of interface circuitry, and is more difficult for software to use.

But! That's not the end of the story. EMM386.EXE continues to play an important role in the typical system, because it can act as gatekeeper for *all* memory located between 640k and 1024k. Thus, peripheral hardware or software components in a system, such as network cards, memory-resident DOS software programs, add-in peripheral cards (such as DSP/analog hardware), etc. end up dealing with EMM386.EXE. For a thorough discussion of how EMM386.EXE affects installation and usage of typical DSP/analog hardware, see "EMM386.EXE Considerations (Reserving Memory Areas)" in section 4.3.2, Memory-Mapped Transfer Gotchas.

System Drivers and Resident Program Considerations

SHARE.EXE

EMM386.EXE

1.4.5 DSP/Analog Hardware Considerations

In cases where a Hypersignal-Macro series package is in use, external drivers provided by the DSP/analog hardware manufacturer or other 3rd party are not required; Hypersignal handles all hardware communication.

If the DSPower-Block Diagram or DSPower-HWLib packages are not being used with a Hypersignal-Macro series package, but in conjunction with drivers supplied by the DSP/analog hardware manufacturer, then these drivers must be compatible with the Windows Multimedia 3.1 standard, including the mmsystem.dll component shipped with Windows 3.1.

DSP/Analog Hardware Considerations Topics

DSP/Analog Hardware Performance Considerations

Using Hardware Manager

Troubleshooting DSP/Analog Hardware Problems

For more information, see also:

1. DSP/Analog Hardware Performance Considerations in section 1.4.2, Performance Considerations.
2. Signalogic DSP Software Users Guide chapter 8, Using Hardware Manager.
3. System Guide section 4.3, Troubleshooting DSP/Analog Hardware Problems.

1.5 Technical Support

At the present time, after-sale technical support is available free-of-charge, on an unlimited basis, using the following methods. If you have read what there is to read, and consulted the chapter 5, Troubleshooting, section, and your problem still doesn't go away, please try the following:

1. send e-mail to:

DSPhelp@signalogic.com

or

72113.1412@compuserve.com

2. send fax to 214-343-0163

3. if absolute urgency is required, contact Signalogic technical support at 214-343-0069

NOTE: Technical support calls to 1-800-DSPower are <u>not</u> accepted.

Notes: